Simply Solving Data Mining

Jilles Vreeken

18 August 2015

UNIVERSITÄT DES SAARLANDES

CLUSTER OF EXCELLENCE

max planck institut informatik

Question of the day

How can we **solve data mining tasks** **without** setting parameters or making assumptions?

Summarisation

Pattern-based modelling is often data summarisation

Good models are characteristic for the data

JMLR

support vector machine
machine learning
state [of the] art
data set
Bayesian network

PRES. ADDRESSES

unit[ed] state[s]
public econ. expenditur
take oath
equal right
exercis power

(Vreeken & Tatti 2012)

What is characteristic?

Pattern-based models are characteristic if, e.g.,

- different data distributions get different models
- different models imply different data distributions

The optimal MDL result
has these properties by definition

In practice, however, we only have approximations…
these properties, however, often hold in practice!

So, what will you solve for us?

clustering, outlier detection, data generation,
distance measures, missing value imputation,
change detection, privacy preservation,
graph clustering, influence propagation,
classification, …

all at an *explorative* angle

few assumptions and parameters
identify interesting **local** structure
describe structure in **simple** terms

Solving data mining tasks

The 'recipe'

- formalise your problem using information theory
- design a (heuristic) algorithm to solve it
- run experiments
- write a paper

Using MDL and pattern-based models

- **formalise** your problem in terms of compression
- **find models** (e.g., code tables) that minimise compressed size

Advantages

- principled
  - i.e., firmly rooted in information theory
- parameter-free
- interpretable
- prior knowledge about task helps to design better encodings

- **excellent results**

- the sky is the limit

Disadvantages

- **very hard problems** to solve
  - hence heuristics, often without guarantees
- **choices, choices**
  - both for the encoding and algorithms

Note: nothing new here, the same as when modelling!

Data, patterns, and models

For simplicity, in the following we consider

- itemset data
- itemsets as patterns
- code tables (as induced by, e.g., KRIMP or SLIM)

Unless noted otherwise

Note, however, that the approaches are generic

- conceptually, at least
- computationally this is not always straightforward

Traditional Data Mining Tasks

Classification using compression

Classification

"The prediction of the class of an object on the basis of some of its attributes."

General recipe

- build a classifier on training data
- assign class labels to (unseen) tuples

How can we do this using compression?

Compression and independence

Assume code table CT and arbitrary transaction t:

$$
\begin{aligned}
L(t \mid CT) &= -\sum_{X \in cover(t|CT)} \log\big(P(X \mid D, CT)\big) \\
&= -\log \prod_{X \in cover(t|CT)} P(X \mid D, CT) \\
&= -\log(Pt \mid D, CT))
\end{aligned}
$$

Note: in the last step, we treat the elements in $cover(t \mid D, CT)$ as if they are independent.

Although we know they are not

(Van Leeuwen, Vreeken & Siebes 2006)

Compression and classification

Assume two databases, $\mathcal{D}_1$ and $\mathcal{D}_2$,
with associated code tables CT_1 and CT_2

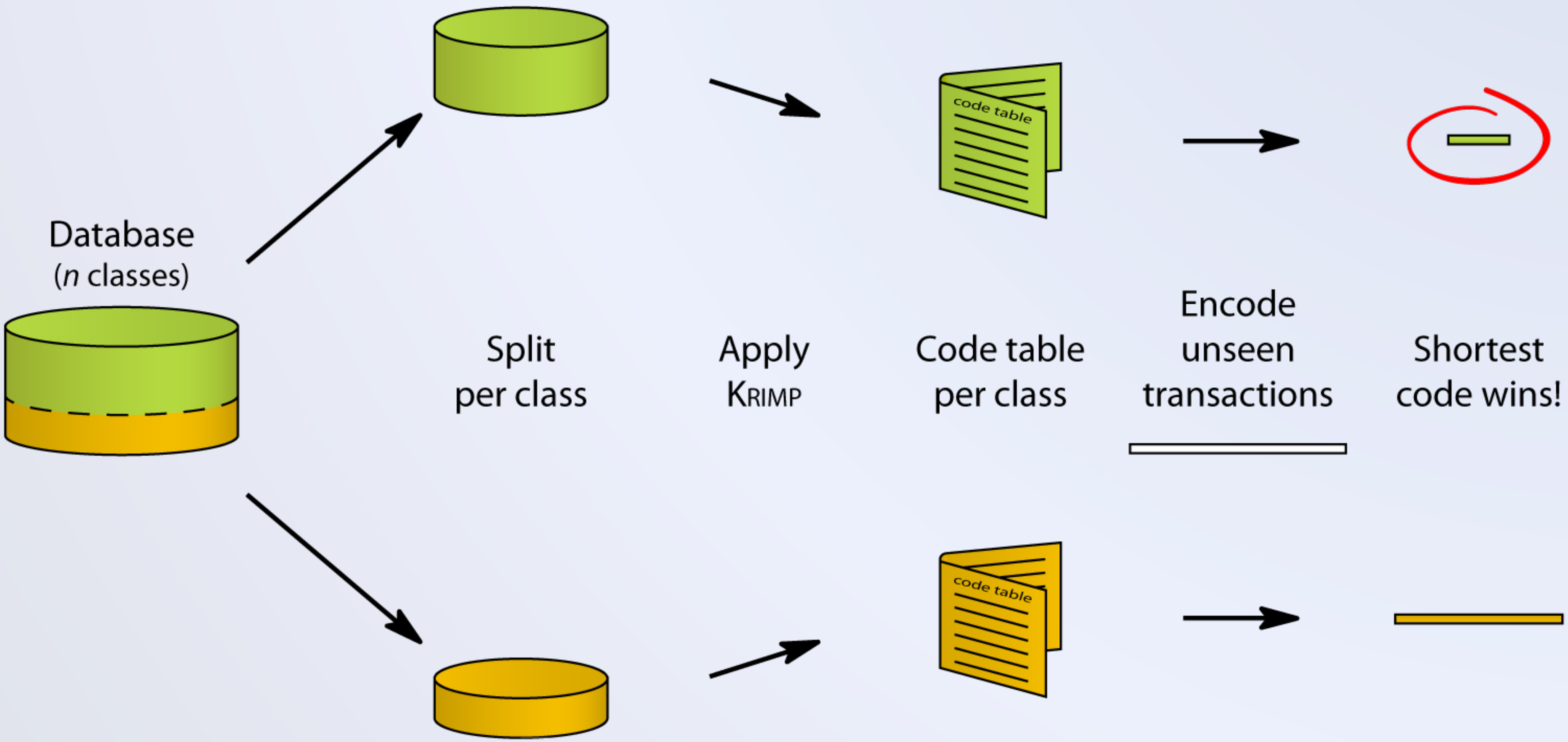
For an arbitrary transaction t

$$L(t \mid CT_1) < L(t \mid CT_2) \leftrightarrow P(t \mid \mathcal{D}_1) > P(t \mid \mathcal{D}_2)$$

Hence, the Bayes optimal choice is to assign t to the database that gives the best compression.

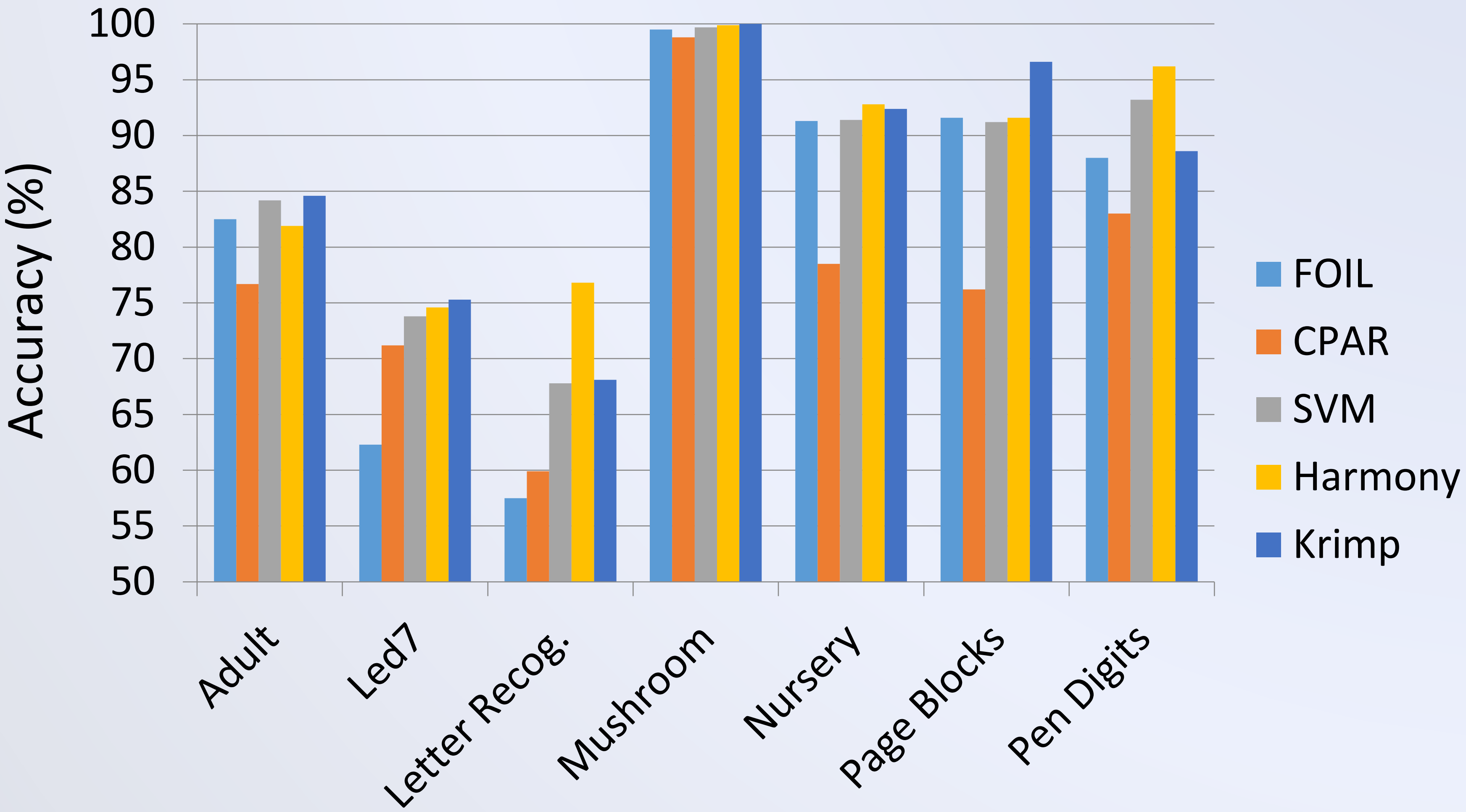
(Van Leeuwen, Vreeken & Siebes 2006)

Compression-based classification



(Van Leeuwen, Vreeken & Siebes 2006)

Classifier performs very well



(Van Leeuwen, Vreeken & Siebes 2006)

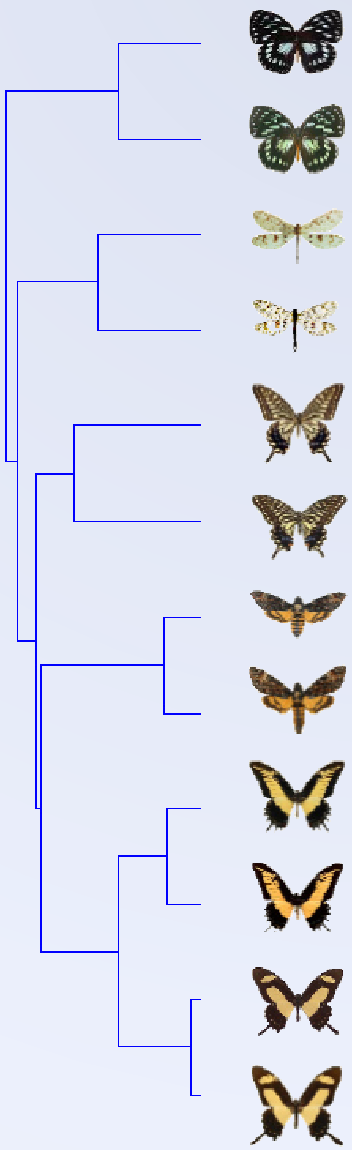
Clustering by compression

Compression and clustering match well

- Normalised Compression Distance (NCD)
  (Cilibrasi & Vitanyi 2005)

They use off-the-shelf compressors,
that **do not use patterns** and thus
their answers are **without explanations**

Often the compressor is **immaterial**!



(Campana & Keogh 2010)

Clustering transaction data

- partition the database into k clusters
- each cluster is characterised by a code table
- no dissimilarity measure required!
- optimal *k* determined by MDL

Formally

Partition $\mathcal{D}$ into $\mathcal{D}_1 \ldots \mathcal{D}_\mathrm{n}$

such that $\sum L(CT_i, \mathcal{D}_i)$

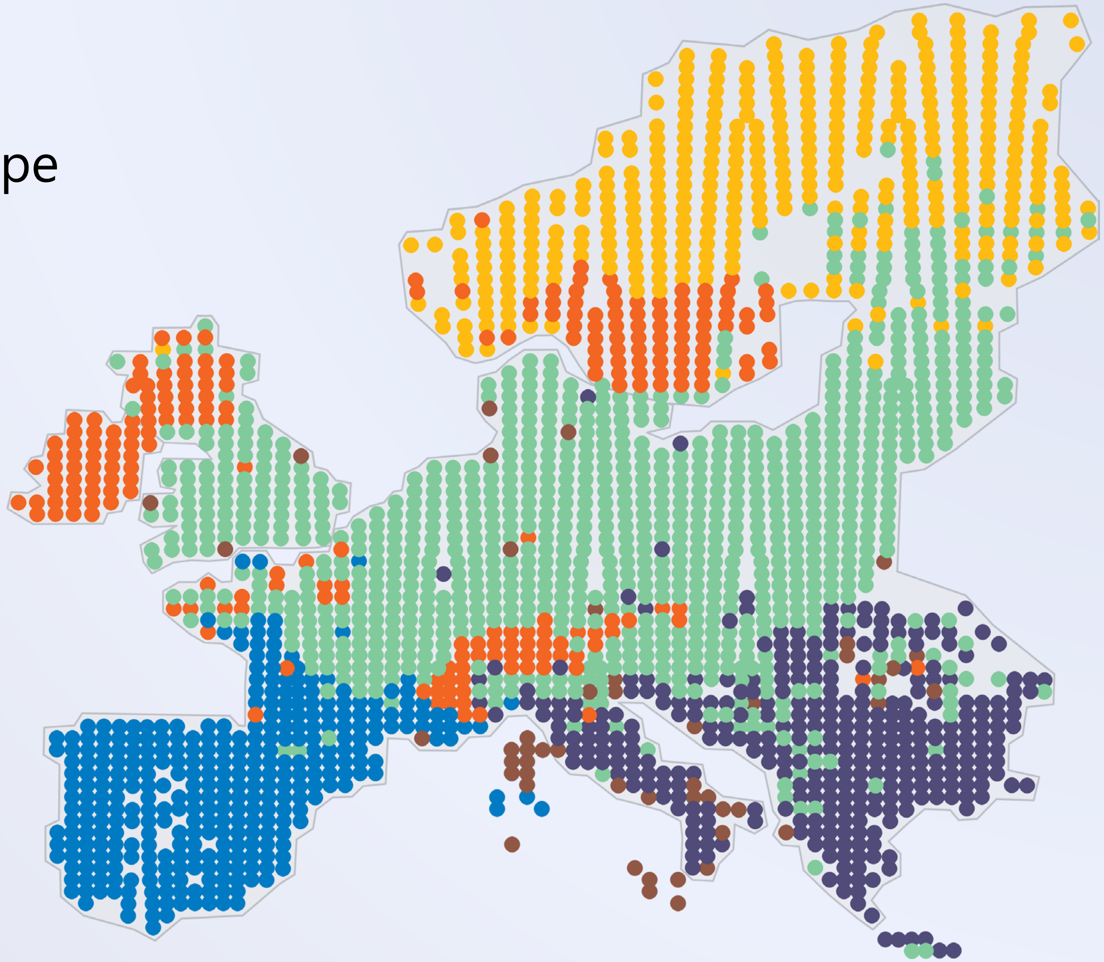
is minimised

(Van Leeuwen, Vreeken & Siebes 2009)

Clustering transaction data

Mammals

- 2221 areas in Europe
- 50x50 km each
- 124 mammals
- *no location info*



$k = 6$, MDL 'optimal'

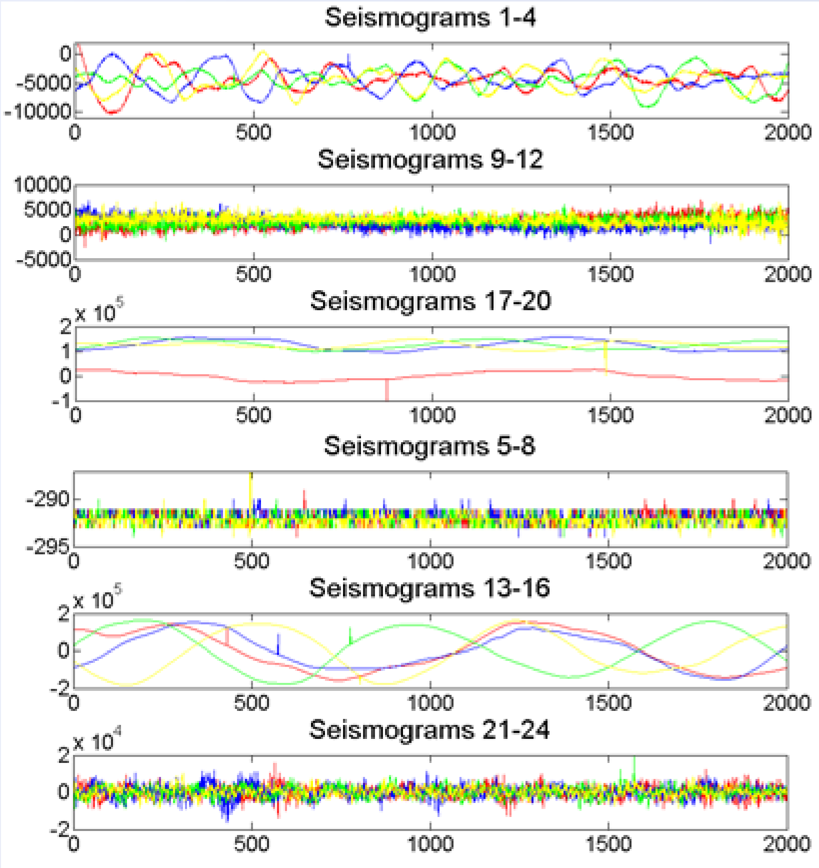
(Van Leeuwen, Vreeken & Siebes 2009)

Clustering seismic data

Time series

- pre-process data using wavelets
- discretise
- patterns span multiple levels

Characterises both old and new events



(Bertens & Siebes 2014)

‘Statistical’ Data Mining Tasks

Differences in data

Suppose we have data from different time periods.

- or: data from multiple branches of a company.

*"What is the **difference**?"*

- can we quantify (dis)similarity between databases?
- what patterns occur more/less over time?
- how typical is an individual transaction for a certain period?

Dissimilarity measure

MDL tells us
the optimal compressor for database x compresses x better than the optimal compressor for database y.

Define $C_x(y)$ as:
the size of database y as compressed by the compressor induced from database x

For all databases x and y, now define the compressor dissimilarity DS as:

$$DS(x,y) = \max\left\{\frac{C_y(x) - C_x(x)}{C_x(x)}, \frac{C_x(y) - C_y(y)}{C_y(y)}\right\}$$

(Vreeken, Van Leeuwen & Siebes 2007a)

Quantifying the difference

Dataset	$\lvert\mathcal{D}\rvert$	#classes	Acc. %	DS between classes	
				min	max
Adult	48842	2	84.6	0.60	
Chess (kr-k)	28056	18	58.0	0.29	2.69
Mushroom	8124	2	100.0	8.24	
Nursery	12960	5	92.4	1.26	10.12
Wine	178	3	97.7	1.27	1.73

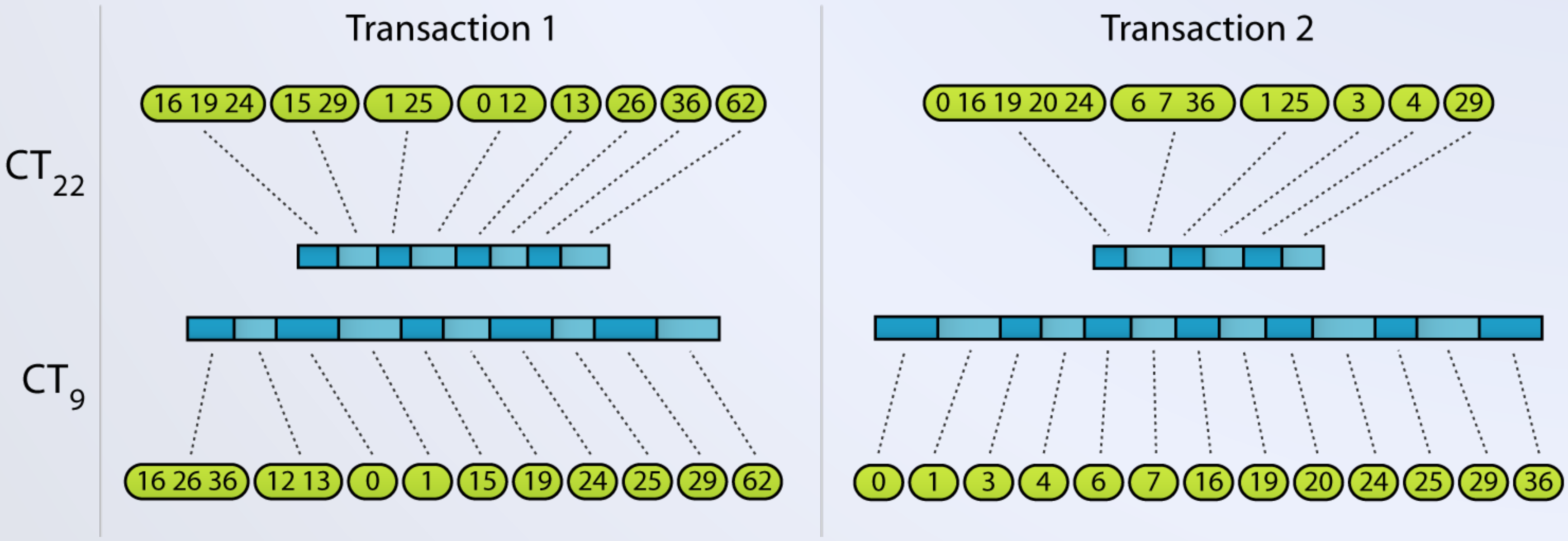
DS is correlated with classification accuracy.

(Vreeken, Van Leeuwen & Siebes 2007a)

Characterising the difference

Encode transactions with compressors induced from different databases.

- Shows recognized patterns, pinpoints differences



(Vreeken, Van Leeuwen & Siebes 2007a)

Data generation

Code tables are characteristic for the data distribution

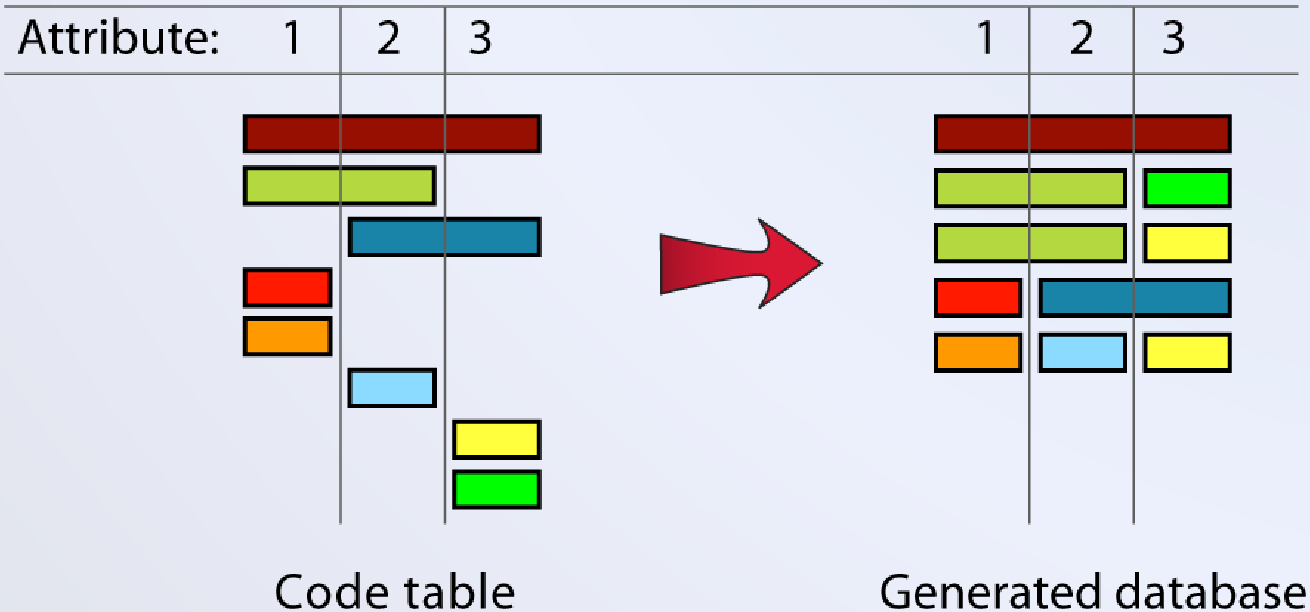
- classification
- dissimilarity quantification
- difference characterisation

Can we generate data from a code table with the same distribution as the original data?

Generating categorical data

Generating a transaction

- Choose a pattern randomly, non-overlapping & weighted by its probability (*code length*)
- Repeat until a value is selected for each attribute



(Vreeken, Van Leeuwen & Siebes 2007b)

Generated data is indistinguishable

Dissimilarity between original and generated data

- dissimilarities between classes range from 0.29 up to 10.12

Dataset	Dissimilarity: Orig vs.	
	Generated	Sample
Chess (kr-k)	0.037	0.104
Iris	0.047	0.158
Mushroom	0.010	0.139
Nursery	0.011	0.045
PenDigits	0.198	0.124

(Vreeken, Van Leeuwen & Siebes 2007b)

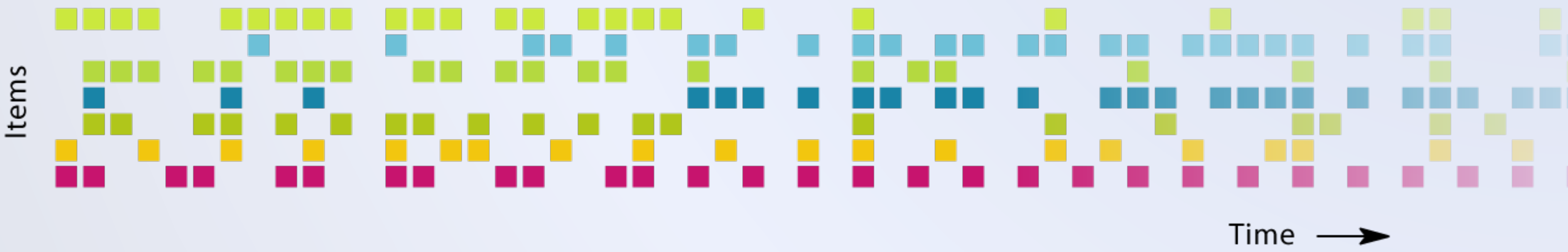
Change in data

*We can **quantify** differences,*
*so can we **detect** them?*

Data streams

- financial world
- sales (supermarkets, online stores, …)
- web

An example data stream



Data stream: a sequence of transactions

(Van Leeuwen & Siebes 2008)

An example data stream

Identify changes in the characteristics of the data

(Van Leeuwen & Siebes 2008)

Change detection in data streams

Partition a finite data stream S
into consecutive substreams $S_1, \ldots, S_k$,
such that the total encoded size

$$\sum L(CT_i, S_i)$$

is minimised.

Streams are not finite.

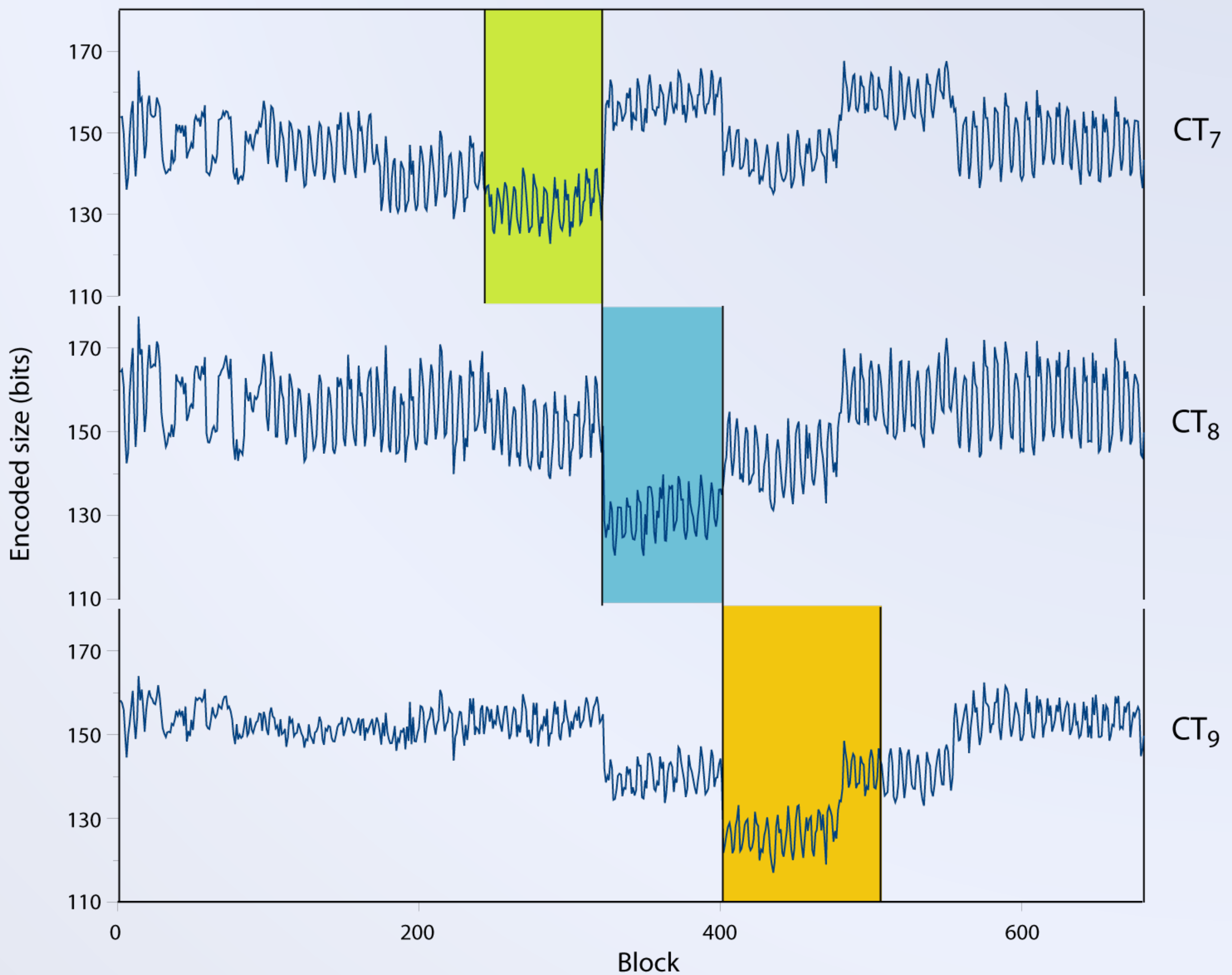
We assume bounded storage and settle for a locally optimal segmentation.

(Van Leeuwen & Siebes 2008)

Accidents

Belgian traffic accidents

- 1991 – 2000
- 340,184 transactions
- 468 items



(Van Leeuwen & Siebes 2008)

The Odd One Out

One-class classification (a.k.a. anomaly detection)

- lots of data for normal situation – insufficient data for target

Compression models the norm

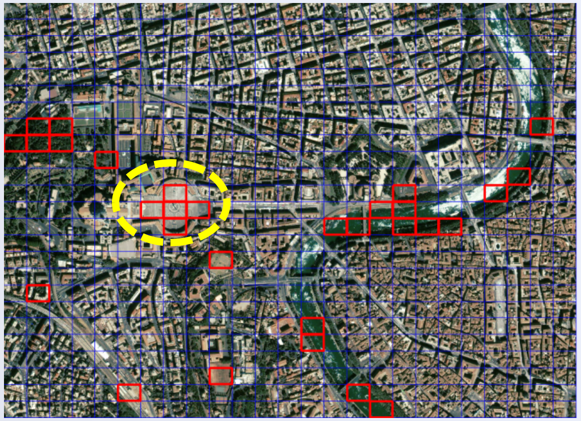
- anomalies will have high description length $L(t \mid CT_{norm})$

Simple, with very nice properties

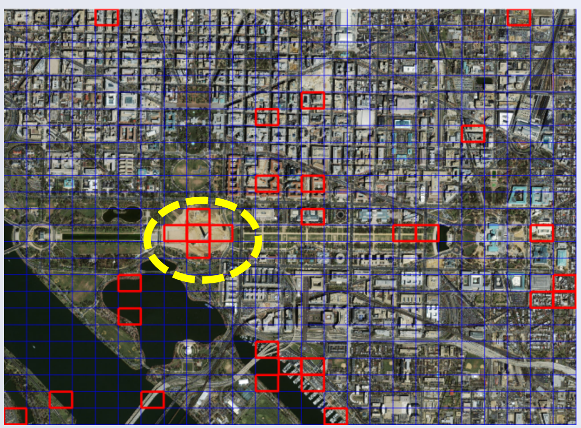
- *performance* high accuracy
- *versatile* no distance measure needed
- *characterisation* '*this part of t is incompressible*'

(Smets & Vreeken 2011, Akoglu et al. 2012)

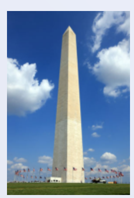
CompreX on images

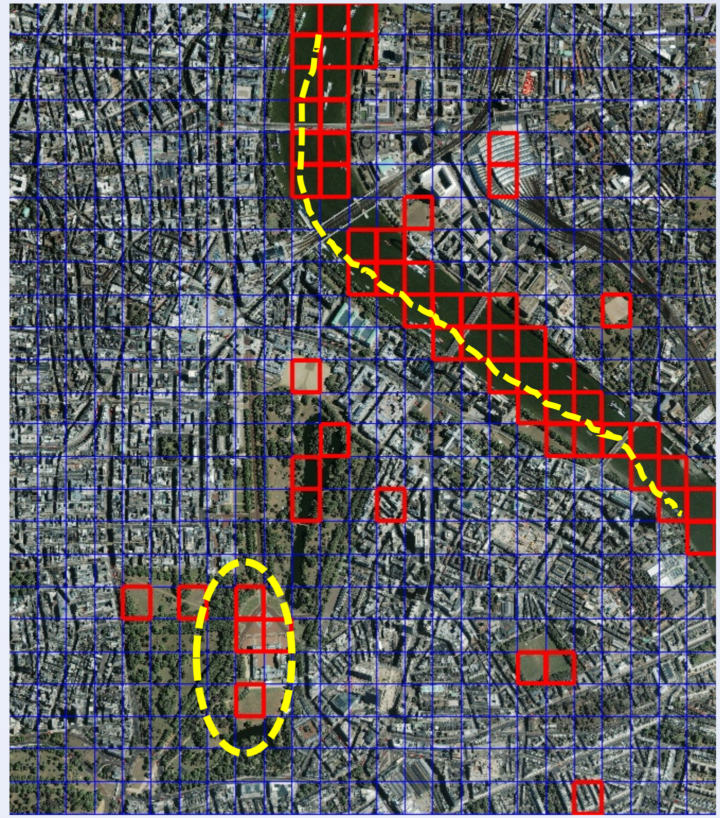


Catholic church, Vatican



Washington Memorial, D.C.





Thames river, Buckingham palace, plain fields, London

(Akoglu et al. 2012)

Filling in the blanks

Use the same principle to get rid of missing values

The completed database that can be
compressed best
is the best completed database.

Good performance explained

- not only global statistics correct
- imputations adhere to the local patterns!

(Vreeken & Siebes 2008)

Novel Mining Tasks in Networks

Application-specific encodings

So far, most applications used generic models

- Pattern-based models that characterise and summarise

Some applications require specific encodings & models

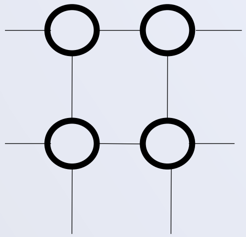
- or, are simply easier to solve with a specific solution

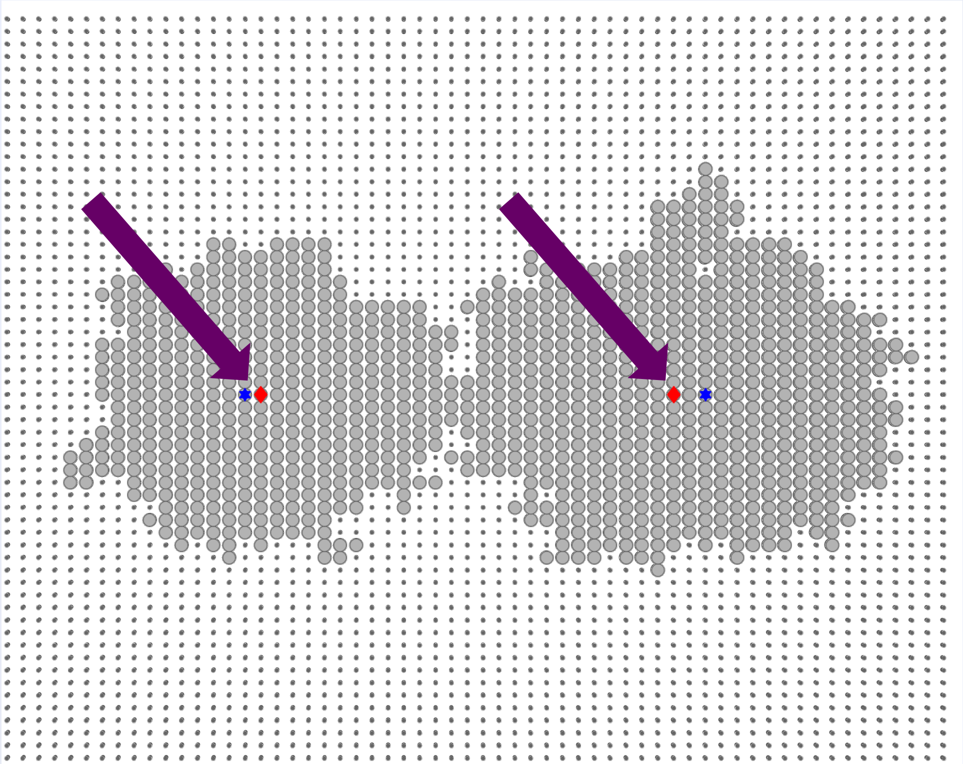
Who are the culprits?

Suppose a graph in which an epidemic spreads

- who caused it?

2-d grid

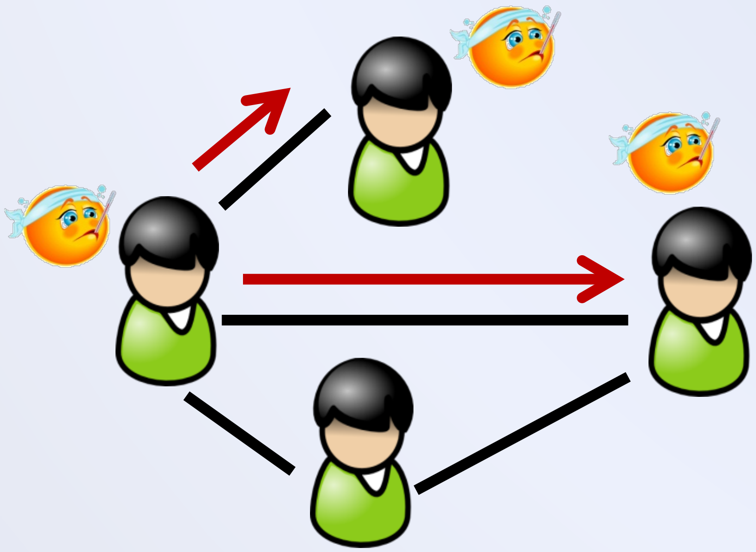




(Prakash, Vreeken & Faloutsos, 2012, 2013)

Virus propagation

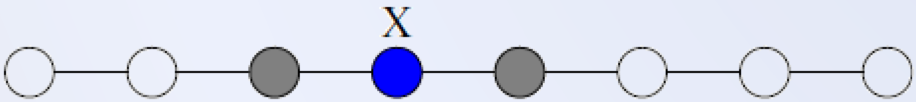
Susceptible-Infected (SI) Model



Diseases over contact networks

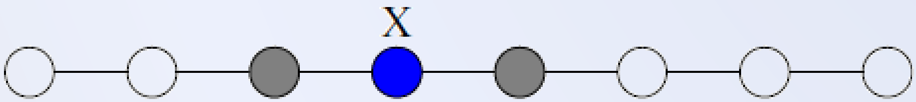
(Prakash, Vreeken & Faloutsos, 2012, 2013)

Culprits: Exoneration

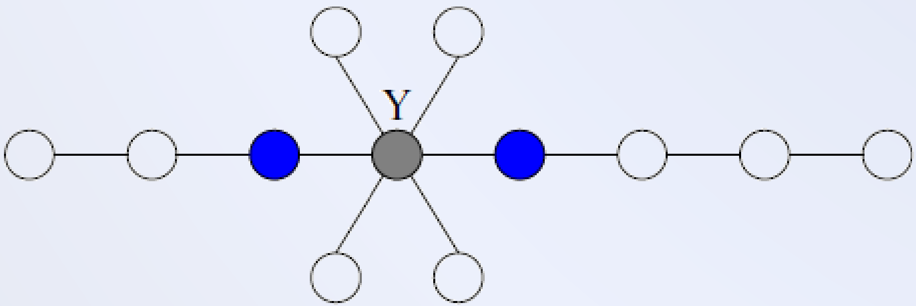


(a) A chain

Culprits: Exoneration



(a) A chain



(b) A chain-star

NetSleuth

Two-part solution

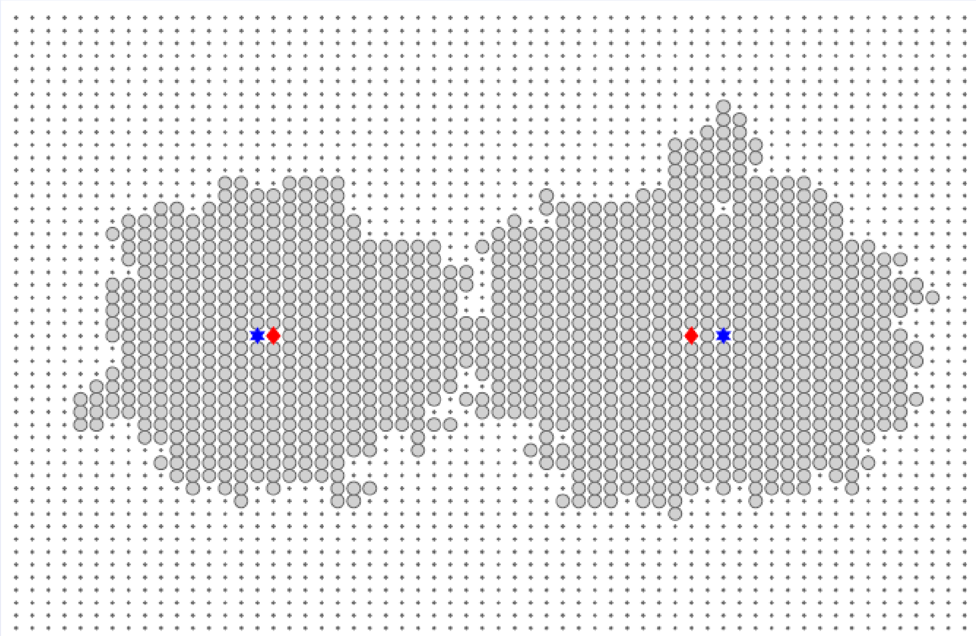
- use MDL for *number* of seeds
- for a given number:
  - exoneration = centrality + penalty

Running time = linear

- in edges and nodes

Solutions found

- more likely to generate snapshot
  than actual seeds!



(Prakash, Vreeken & Faloutsos, 2012, 2013)

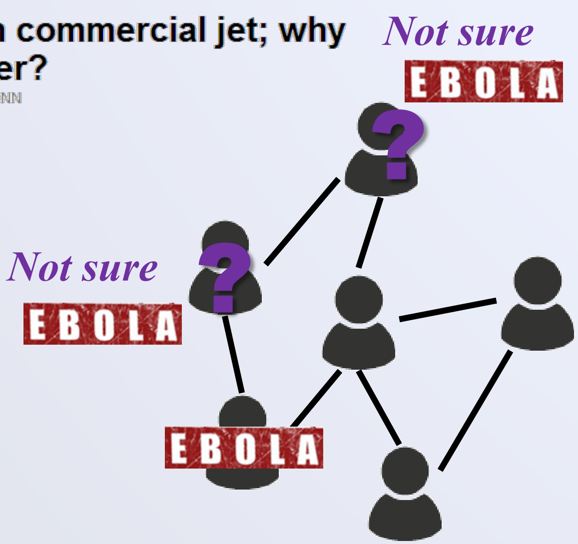
But: Real data is noisy!

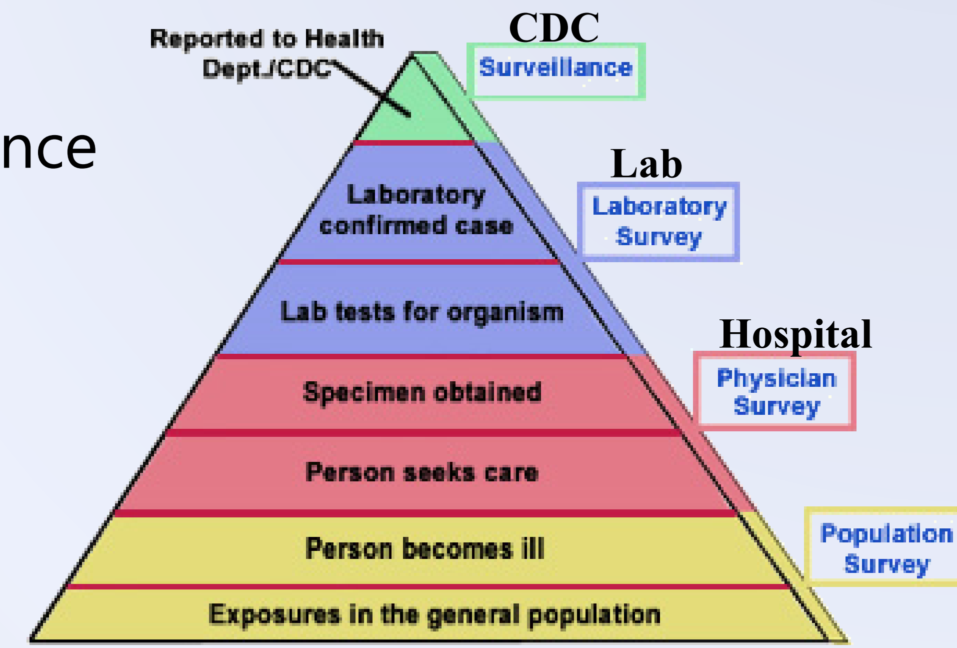
We don’t know who exactly are infected

- Epidemiology
  - Public-health surveillance


BREAKING NEWS
Official: Nurse with Ebola called CDC before flying
Ebola patient flew on commercial jet; why didn't anyone stop her?
By Catherine E. Shoichet, Josh Levs and Holly Yan, CNN
updated 11:32 PM EDT, Wed October 15, 2014

CNN headlines





Surveillance Pyramid
[Nishiura+, PLoS ONE 2011]

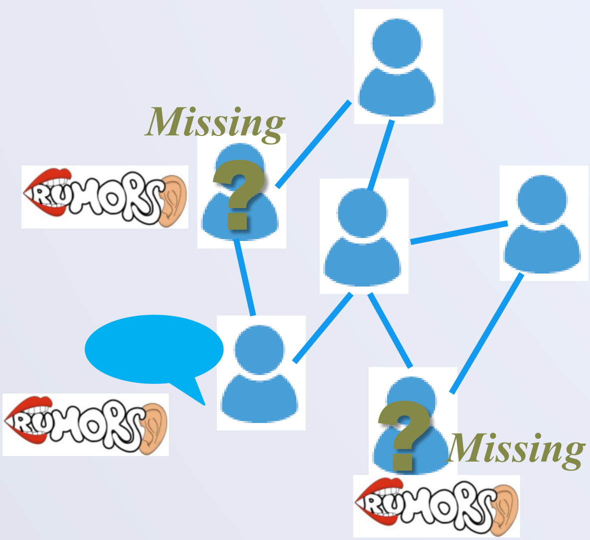
Each level has a certain probability to miss some truly infected people

Real data is noisy!

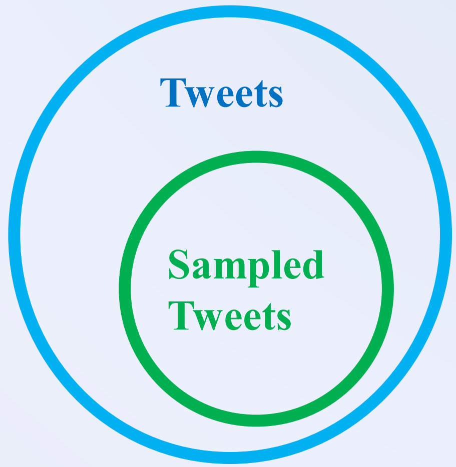
***Correcting missing data** is by itself very important*

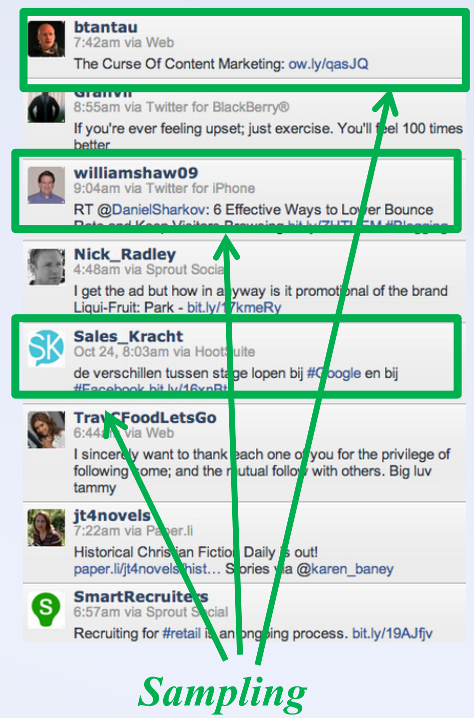
Social Media

- Twitter: due to the uniform samples [Morstatter+ 2013], the relevant ‘infected’ tweets may be missed



DATASIFT

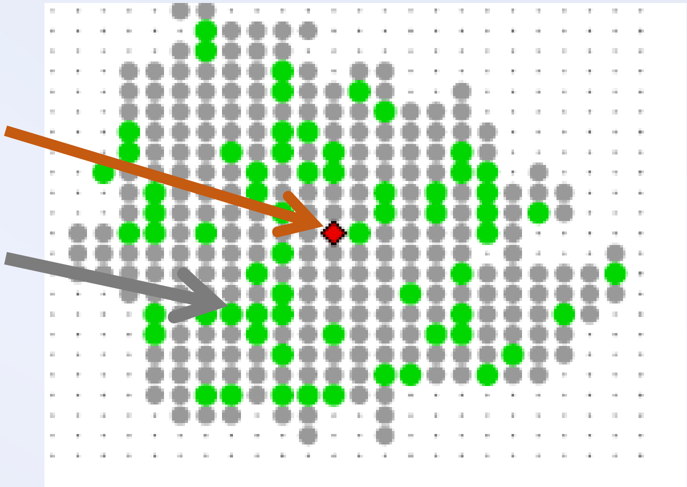




The NetFill Problem

- GIVEN:
  - Graph $G(V,E)$ from historical data
  - Infected set $D \subset V$, sampled ($p\%$) and incomplete
  - Infectivity β of the virus (assumed to follow the SI model)

- FIND:
  - Seed set i.e. patient zeros/culprits
  - Set C^{-} (the missing *infected* nodes)
  - Ripple R (the order of infections)



(Sundareisan, Vreeken & Prakash, 2015)

Model (S, R) Cost

How to score a seed set ($\mathcal{S}$)

$$\mathcal{L}(\mathcal{S}) = \mathcal{L}_{\mathbb{N}}(|\mathcal{S}|) + \log \binom{N}{|\mathcal{S}|}$$

Encoding integer $|\mathcal{S}|$

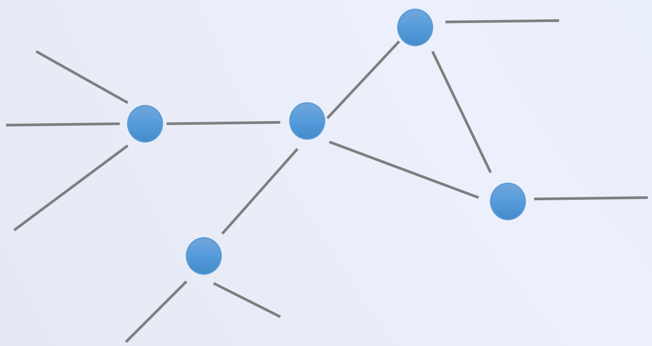
Number of possible $|\mathcal{S}|$-sized sets

How to score the ripple?

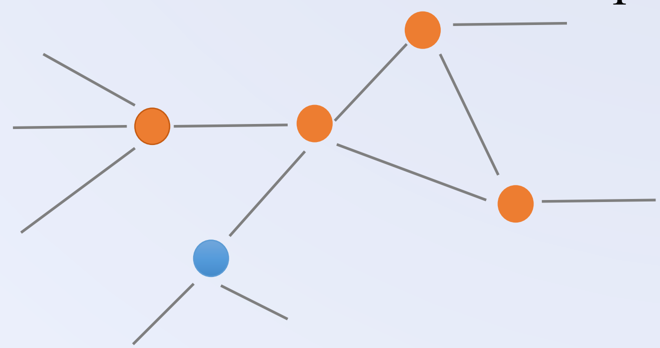
Model (S, R) Cost

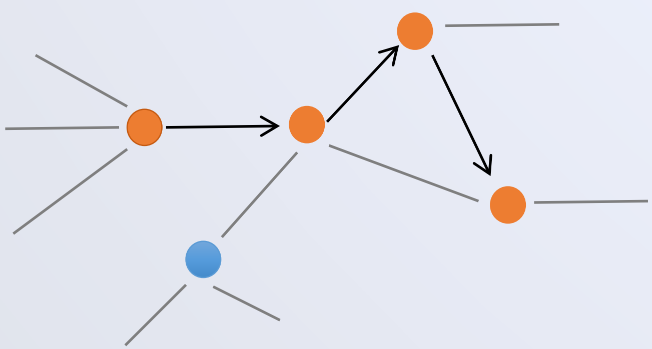
Scoring a ripple (R)

Original Graph

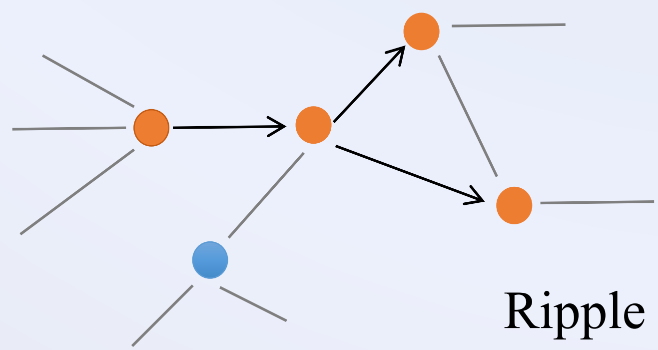


Infected Snapshot





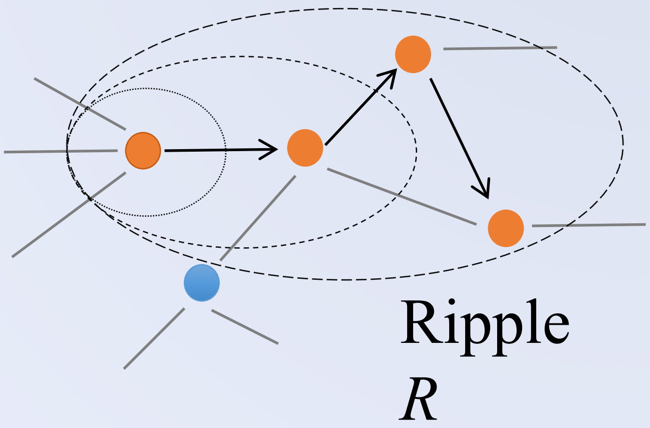
Ripple R_1



Ripple R_2

Model (S, R) Cost

Ripple cost



$$\mathcal{L}(R \mid \mathcal{S}) = \mathcal{L}_{\mathbb{N}}(T) + \sum_{t}^{T} \mathcal{L}(\mathcal{F}^t)$$

How long is the ripple

How the ‘frontier’ advances

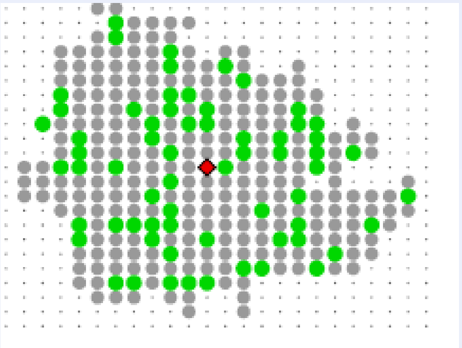
Cost of the data (C-)

Now you know too much – for you to know what was D we need to transmit which are the missed nodes C^- (green nodes)

$$\mathcal{L}(C^- \mid \gamma) = -\log \Pr(|C^-| \mid \gamma) + \log \binom{|I|}{|C^-|}$$

$$\text{with } \Pr(|C^-| \mid \gamma) = \binom{|I|}{|C^-|} \gamma^{|C^-|} (1-\gamma)^{|I|-|C^-|} \quad ,$$

Detail: $\gamma = 1 - p$ i.e. the probability of a node to be truly missing



Total MDL Cost

Finally, we have

$$L(D, \mathcal{S}, R) = L(\mathcal{S}) + L(\,R \mid \mathcal{S}\,) + L(D \mid \mathcal{S}, R)$$

Our problem is now to find those $\mathcal{S}, R, C^{-}$ that minimize it

Our Approach: Decoupling

The two problems are

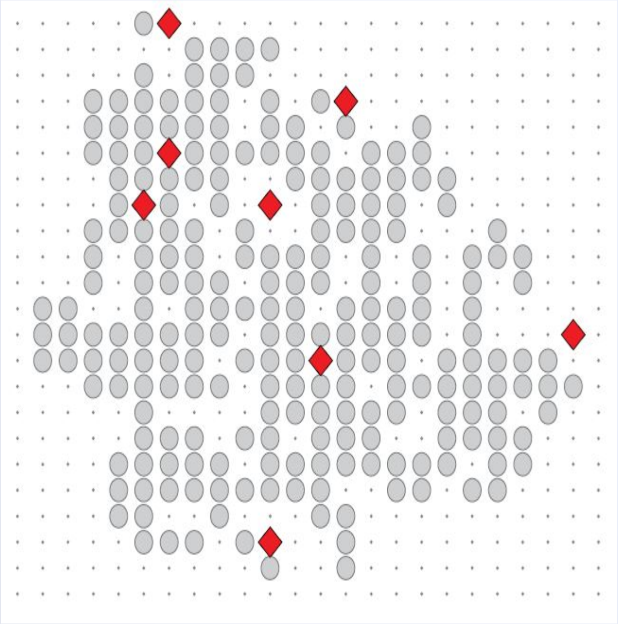
1) finding the seeds and ripple $(\mathcal{S}, R)$

2) finding the missing nodes (C^{-})

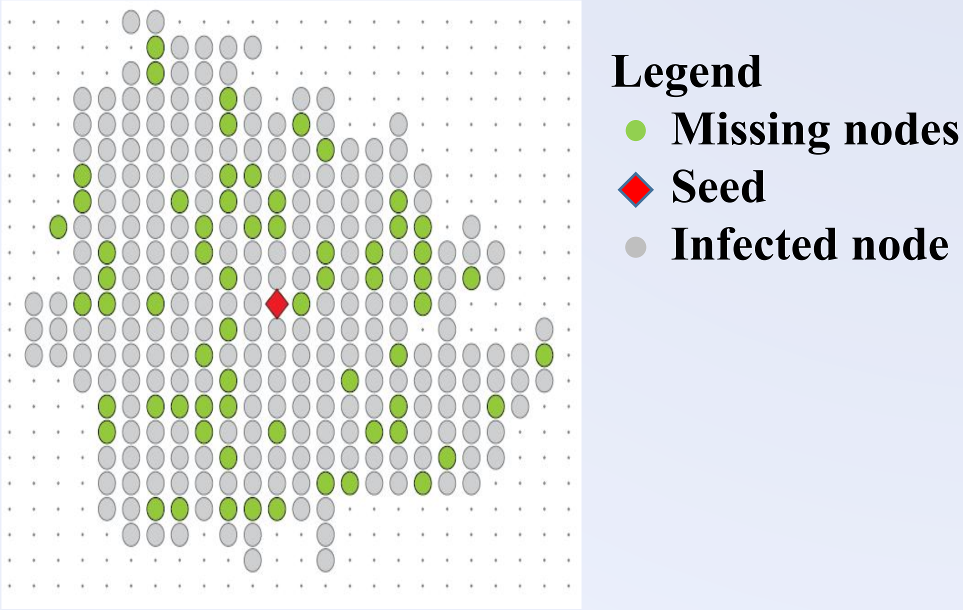
Can we decouple these problems?

Decoupling the problems (contd.)

Finding **seeds depends on missing** nodes.



NetSleuth:
no missing nodes as input,
no missing nodes as output

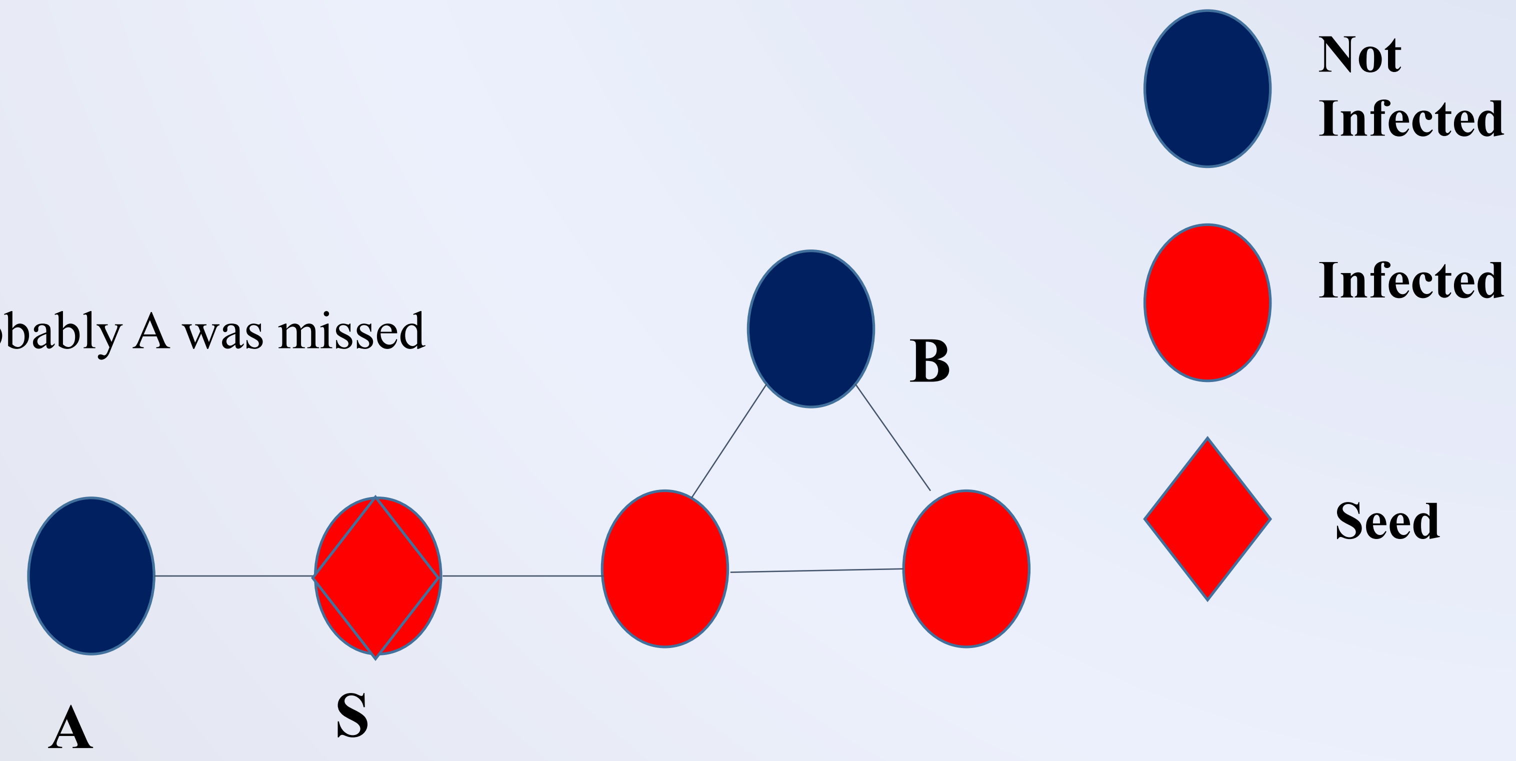


NetFill:
correctly fills in the
nodes missing from input

Decoupling the problems (cont.)

Finding **missing nodes also depends on seeds**.

Most probably A was missed



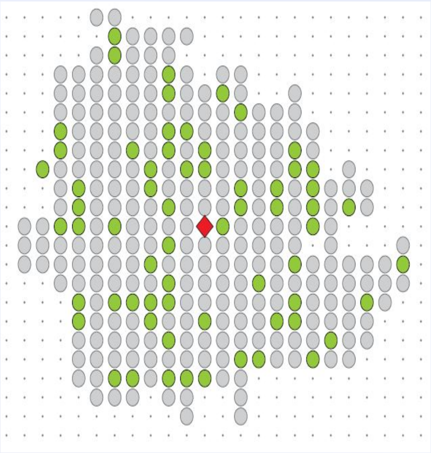
Finding missing nodes ($\mathbb{C}^-$) and culprits ($\mathcal{S}$)

1) Suppose an oracle gives us the missing nodes (C^-)
2) We have complete infected set ($D \cup C^-$)
3) Apply NetSleuth directly

NO SAMPLING INVOLVED

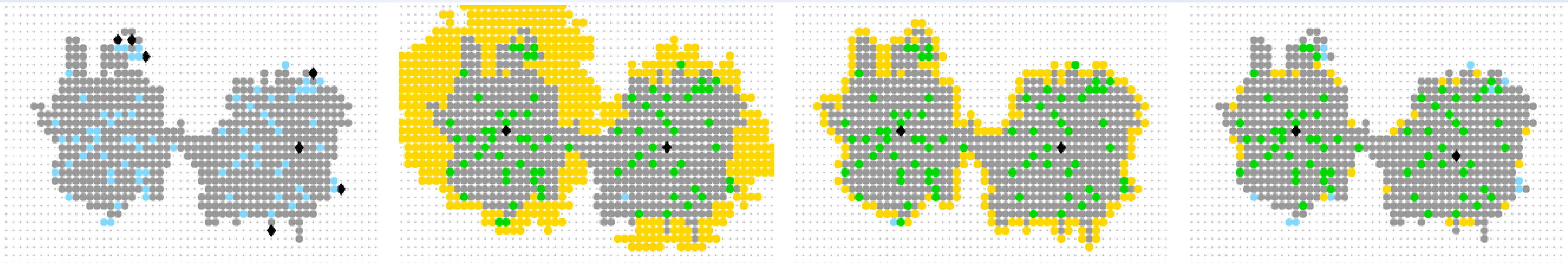
And will give us the seed set!

Applying NetSleuth* on Oracle's Answer



Legend
- **Missing nodes**
- **Seed**
- **Infected node**

Visualizing Performance (Grid connected)



NetSleuth
Seeds
Missing nodes

SIMULATION
Seeds
Missing nodes

FRONTIER
Seeds
Missing nodes

NETFILL
Seeds
Missing nodes

Legend:
Correct **FP** **FN** **Seeds** **Infected**

Meme-Tracker– case study

96,000 node graph for the meme "State of the economy"

What did we find?

Truly missing websites!

Examples include "www.nbcbayarea.com", "chicagotribune.com" and some blog posts.

NBC

Chicago Tribune

Given a 'list' of authors...

What can we say?

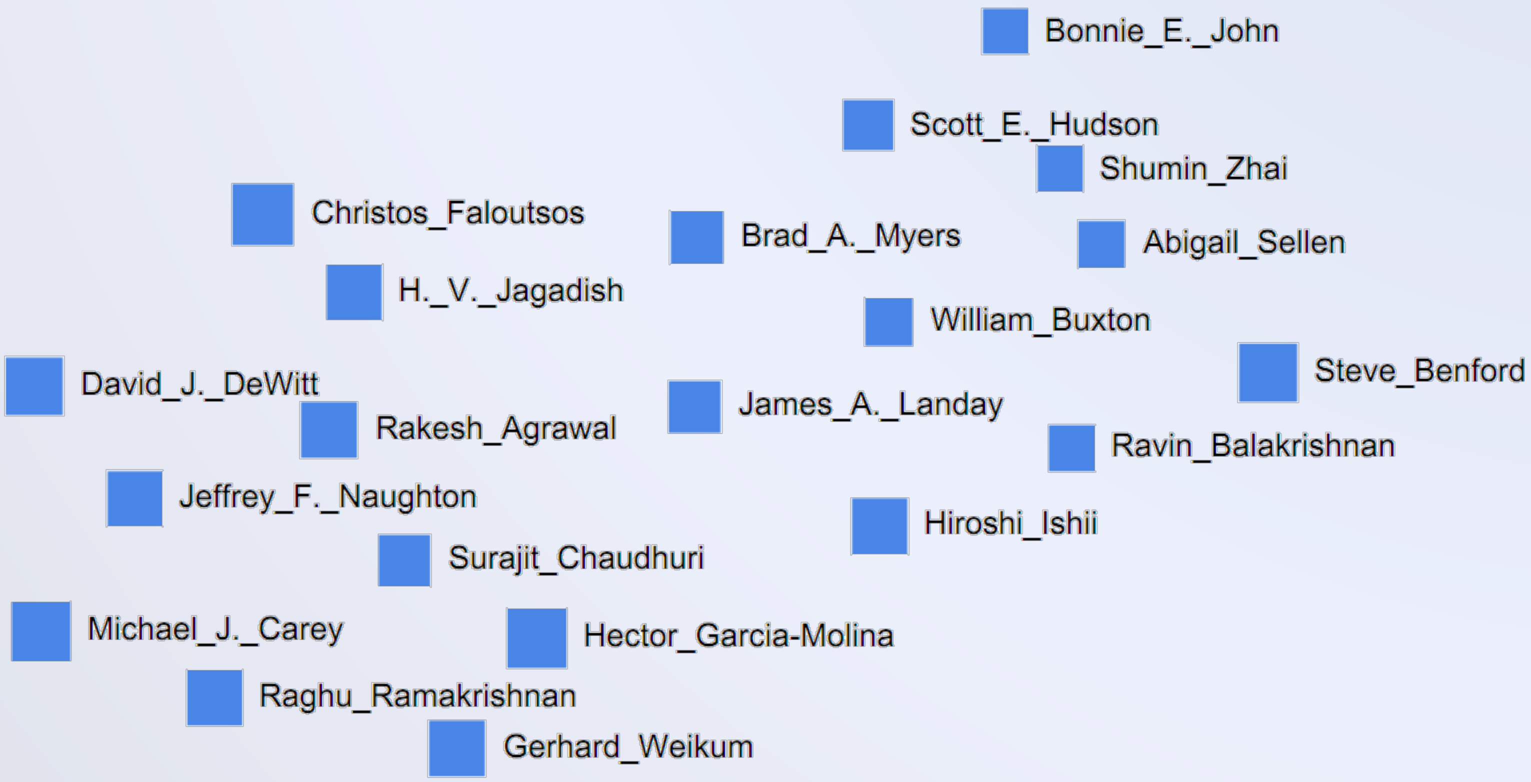
- Christos Faloutsos
- H. V. Jagadish
- David J. DeWitt
- Bonnie E. John
- Hector Garcia Molina
- James A. Landay
- Brad A. Myers
- Jeffrey F. Naughton
- Hiroshi Ishii
- Gerhard Weikum
- William Buxton
- Raghu Ramakrishnan
- Michael J. Carey
- Rakesh Agrawal
- Surajit Chaudhuri
- Scott E. Hudson
- Shumin Zhai
- Abigail Sellen
- Steve Benford
- Ravin Balakrishnan

(Akoglu et al. 2013)

Given a 'list' of authors...

What can we say?

- let's use relational information

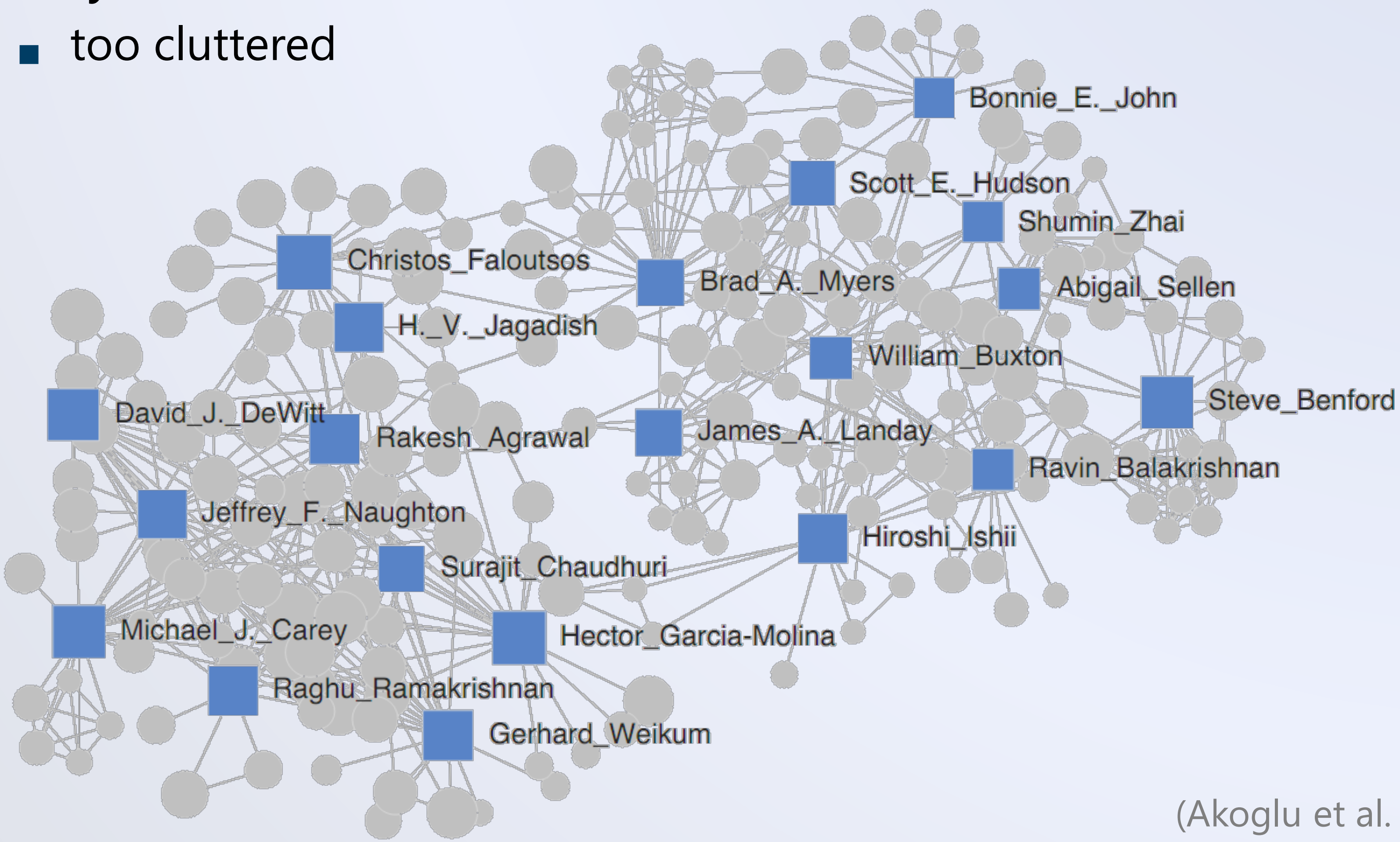


(Akoglu et al. 2013)

Using the co-authorship graph…

Any structure?

- too cluttered



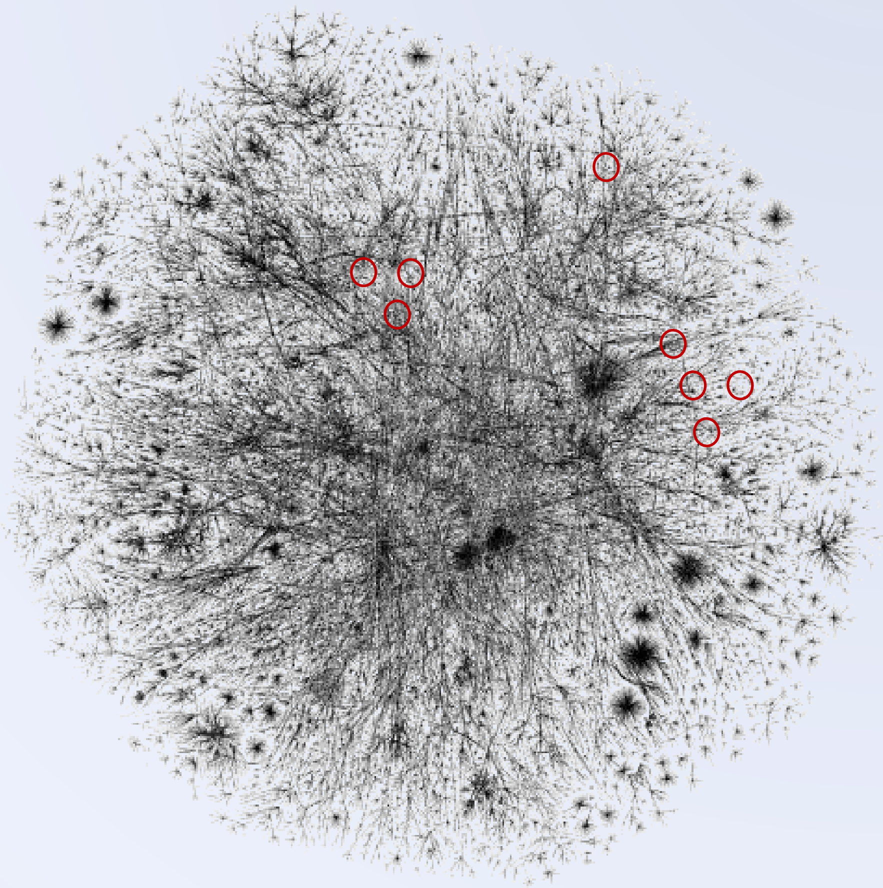
(Akoglu et al. 2013)

The Problem

Given
- a **large graph G**
- a **handful of nodes S** marked by an external process

What can we say about **S**?
- are they **close by**?
- are they **segregated**?
- do they form **groups**?

Can we connect them?
- with **simple** paths?
- maybe using a few **connectors**?

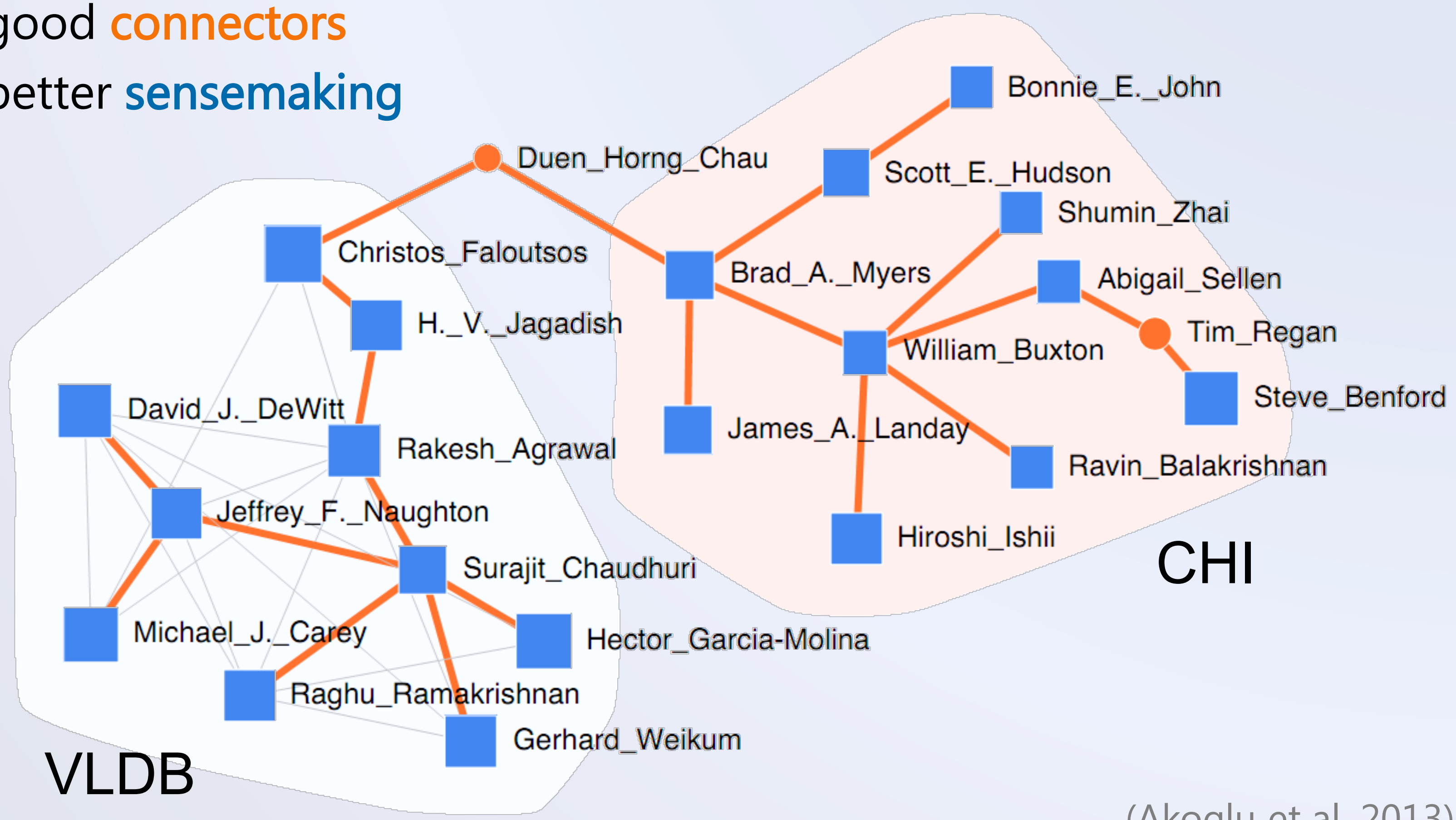


(Akoglu et al. 2013)

Example

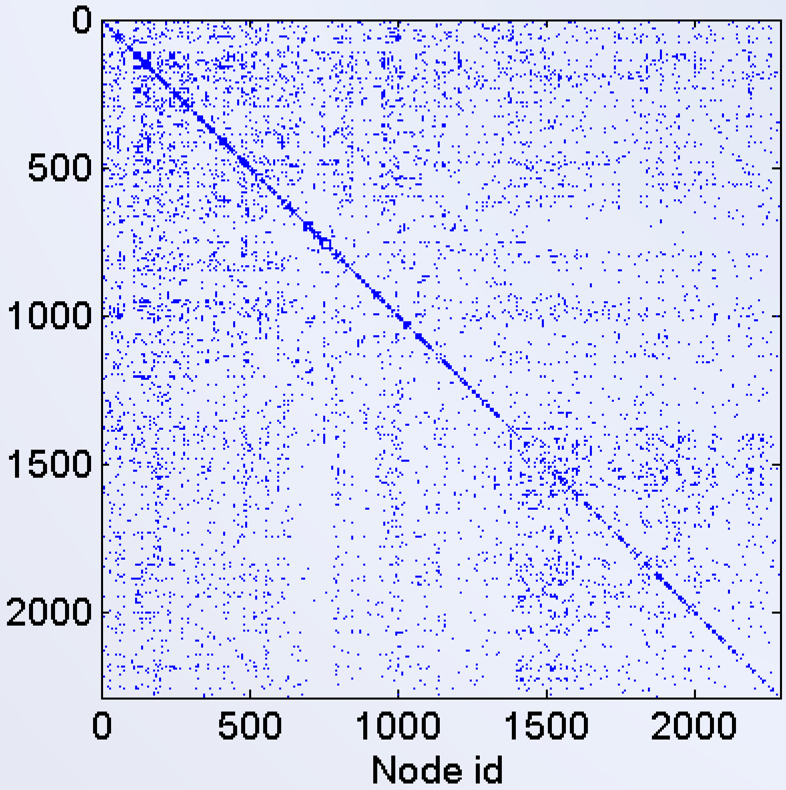
Simple connection pathways

- good connectors
- better sensemaking

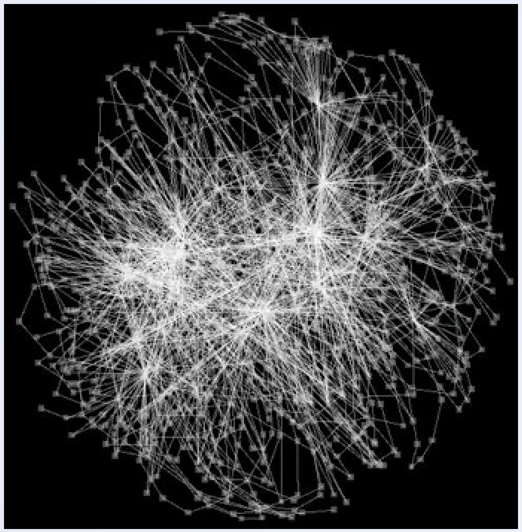


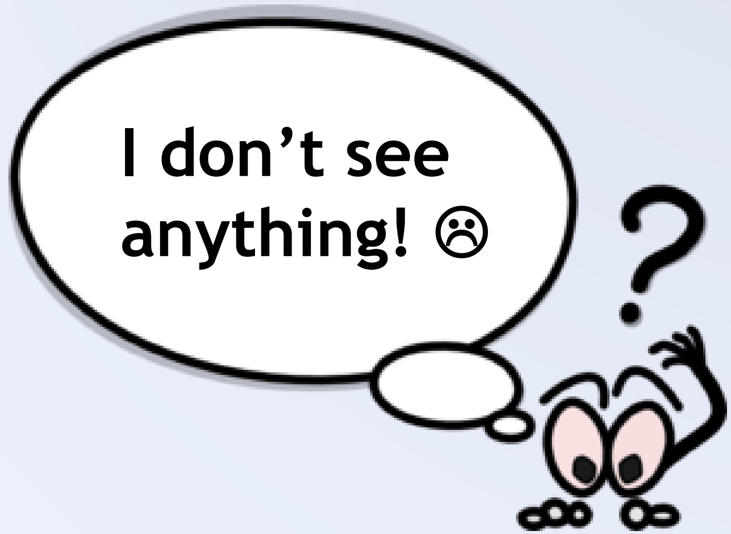
(Akoglu et al. 2013)

Staring at an Adjacency Matrix



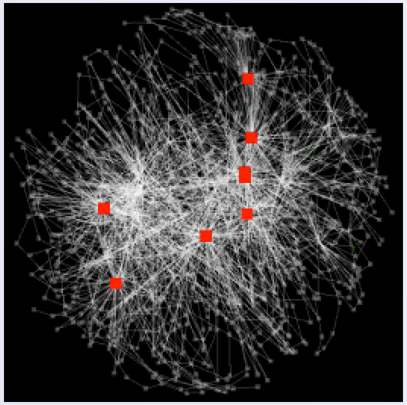
Staring at a Hairball

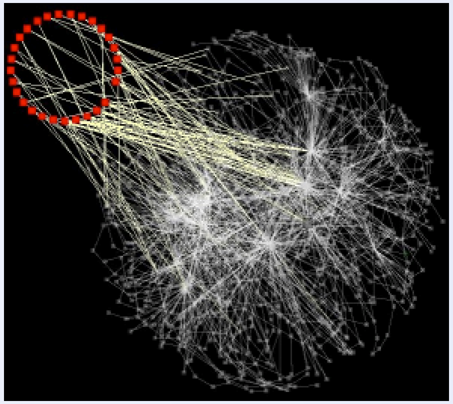


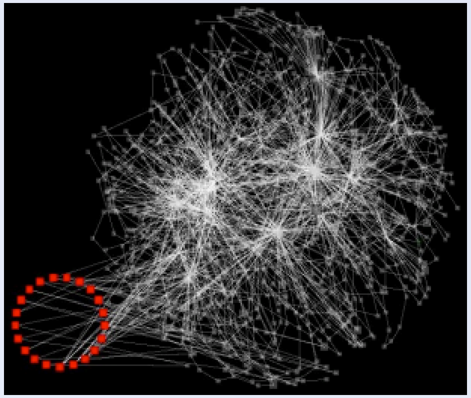


Nodes: wiki editors
Edges: co-edited

Example: Wikipedia Controversy







Stars:
admins,
bots,
heavy users

Bipartite cores: edit wars

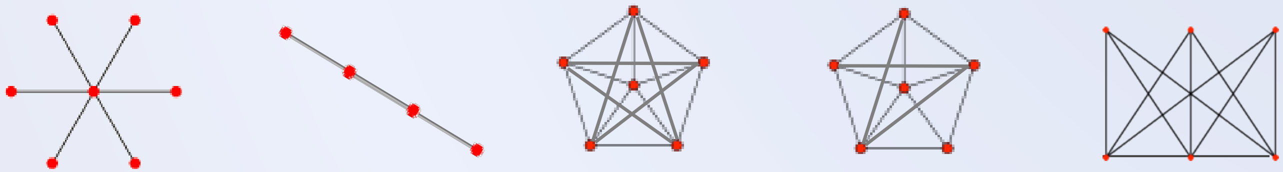
Kiev vs. Kyiv

vandals

Nodes: wiki editors
Edges: co-edited

VoG: Main Idea

1) Use a graph vocabulary:



2) Best graph summary

➔ optimal compression (MDL)

(Koutra et al. 2014, 2015)

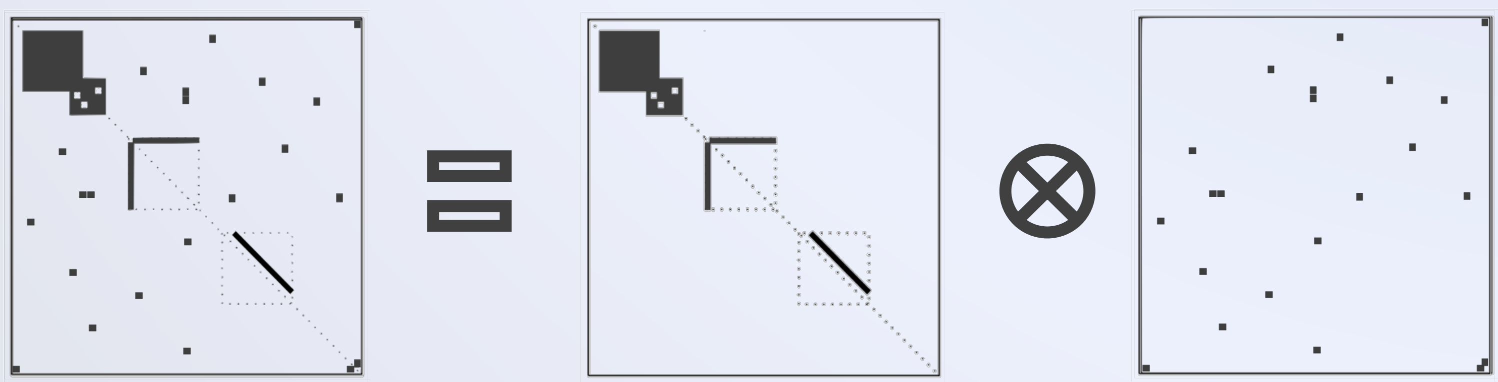
Minimum Graph Description

Given: - a graph G with adjacency matrix A
- vocabulary Ω

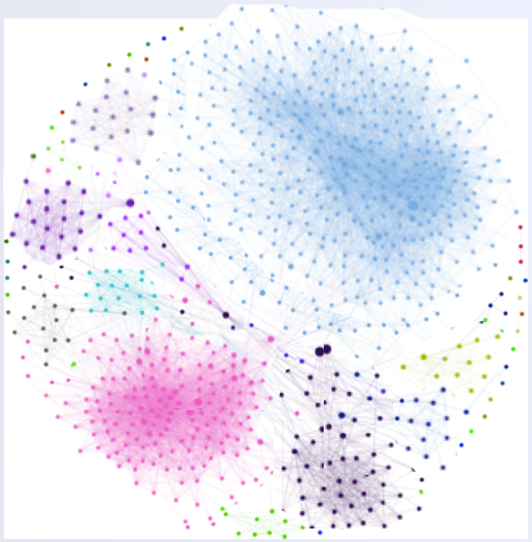
Find: model M s.t.

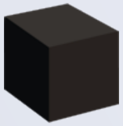
$$L(G, M) \;=\; \min L(M) \,+\, L(E)$$

Adjacency A **Model** M **Error** E



Step 1: Graph Decomposition





We ***can*** **use**:

Any decomposition method

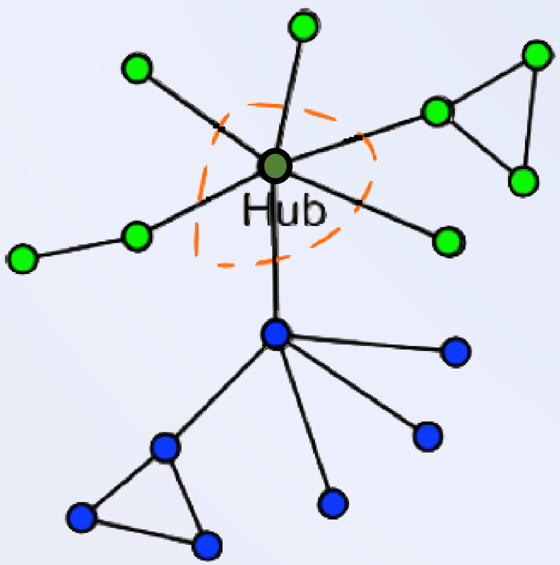
We did use/adapt:

SLASHBURN

(Kang & Faloutsos, 2011)

SnB Graph Decomposition

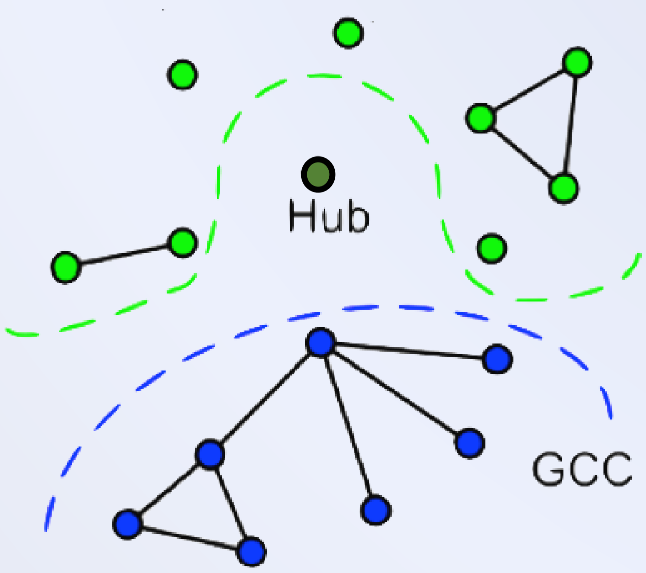
Slash top-k hubs, *burn* edges



Before

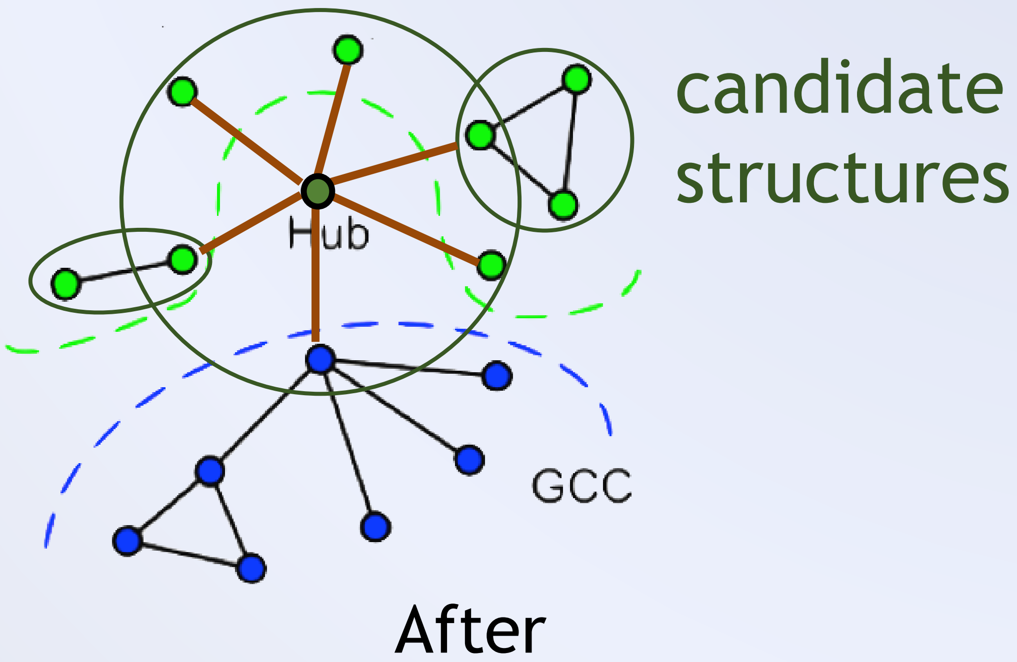
SnB Graph Decomposition

Slash top-k hubs, *burn* edges



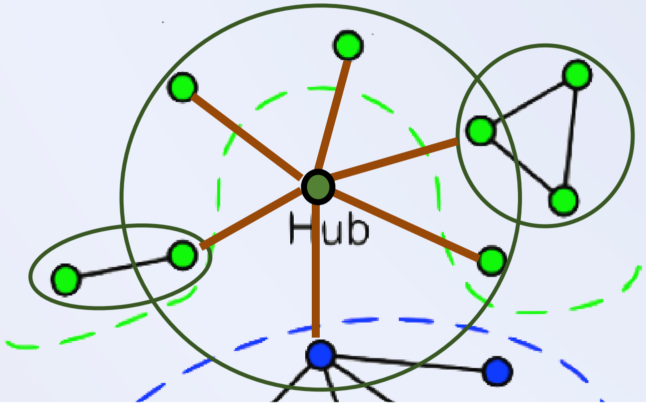
SnB Graph Decomposition

Slash top-k hubs, *burn* edges



SnB Graph Decomposition

Slash top-k hubs, *burn* edges

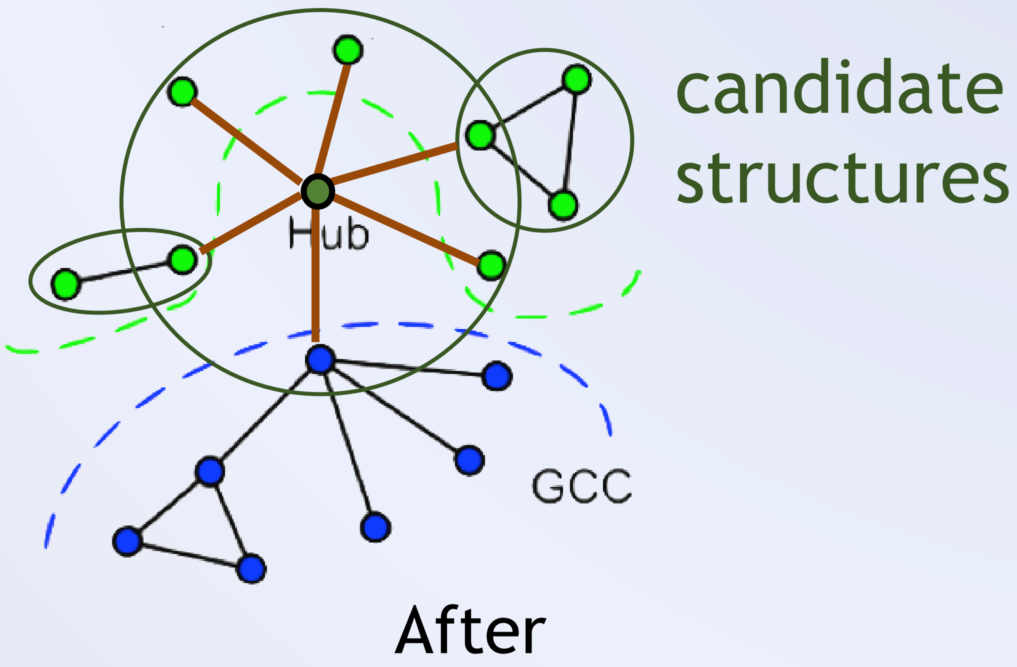


candidate structures

Notice that the structures *can overlap*!

SnB Graph Decomposition

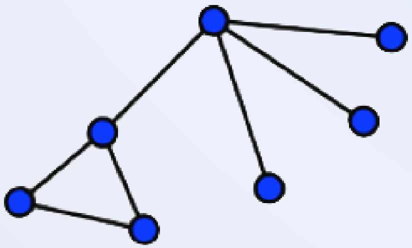
Slash top-k hubs, *burn* edges



SnB Graph Decomposition

Slash top-k hubs, *burn* edges

Repeat on the remaining GCC

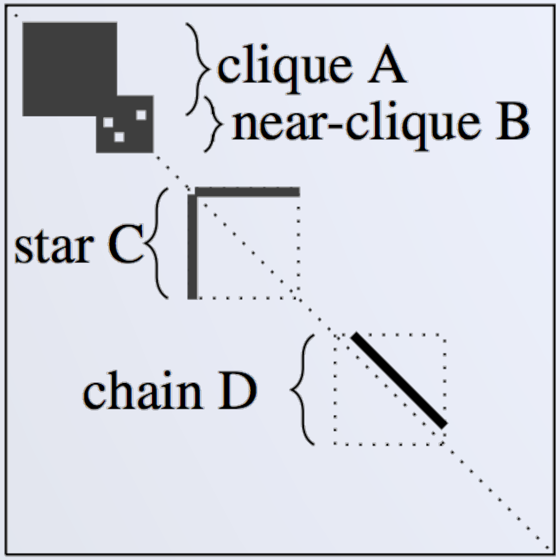


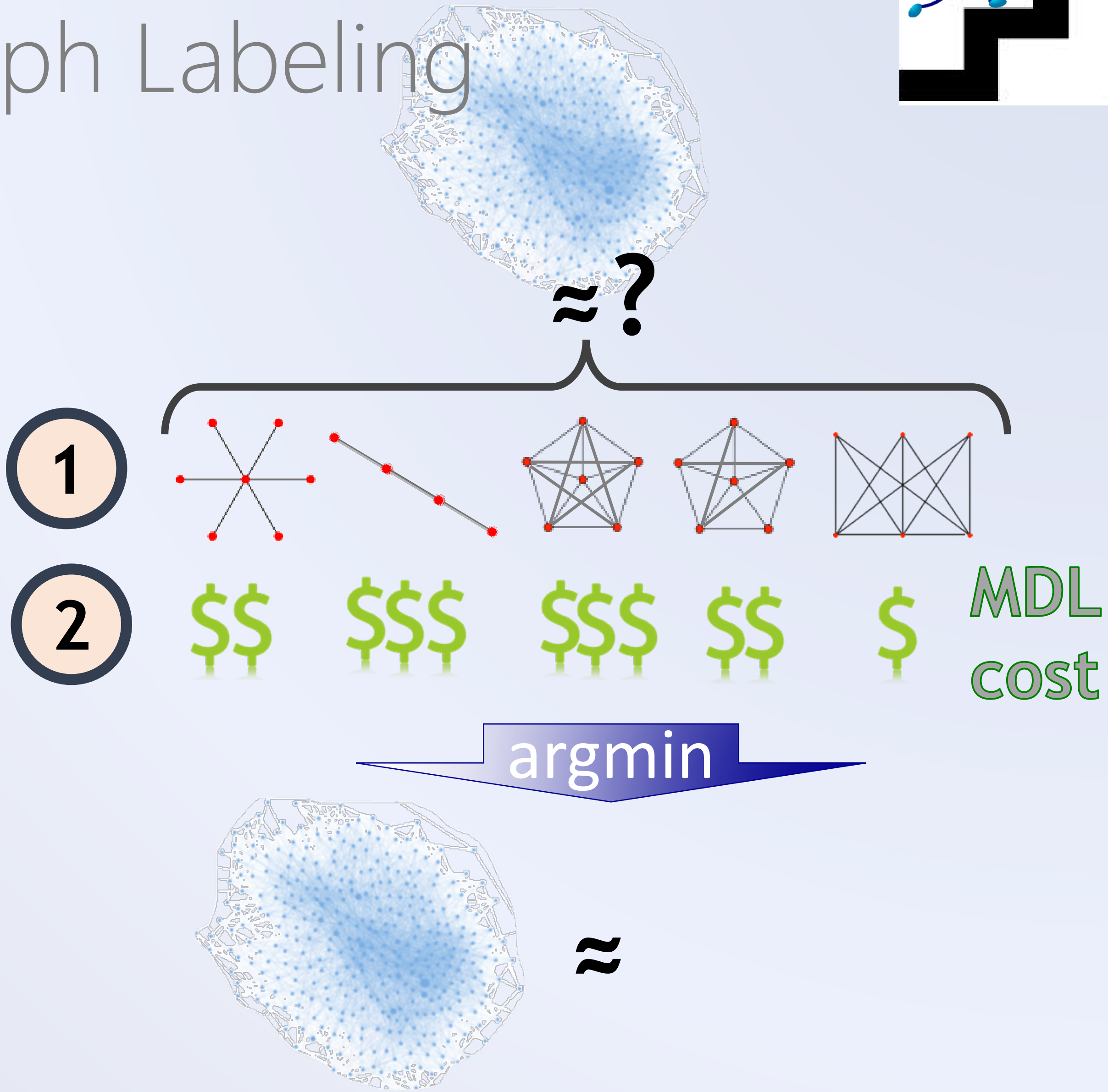
GCC

We got candidate structures.

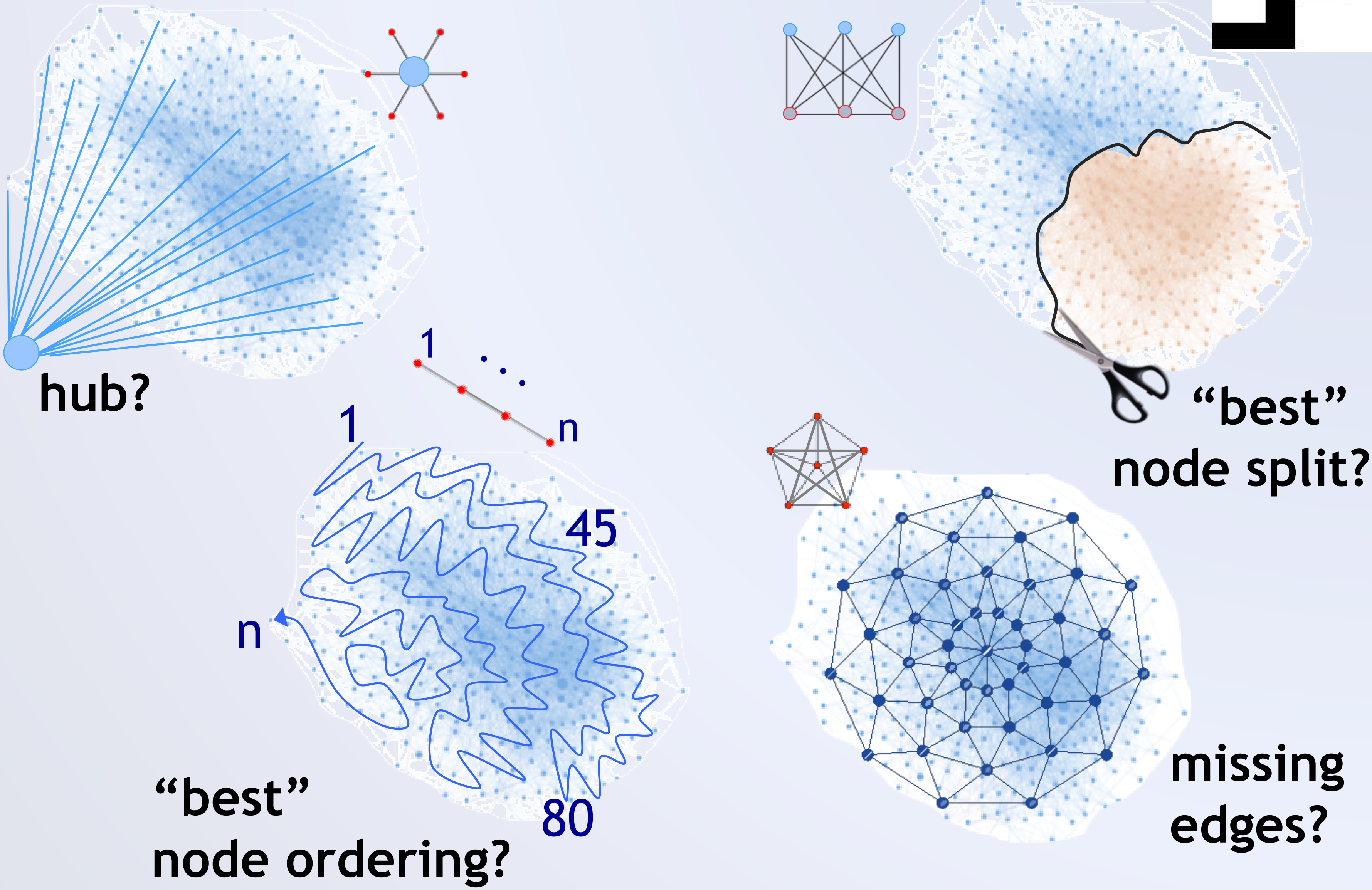
Now, how can we 'label' them?

Step 2: Graph Labeling





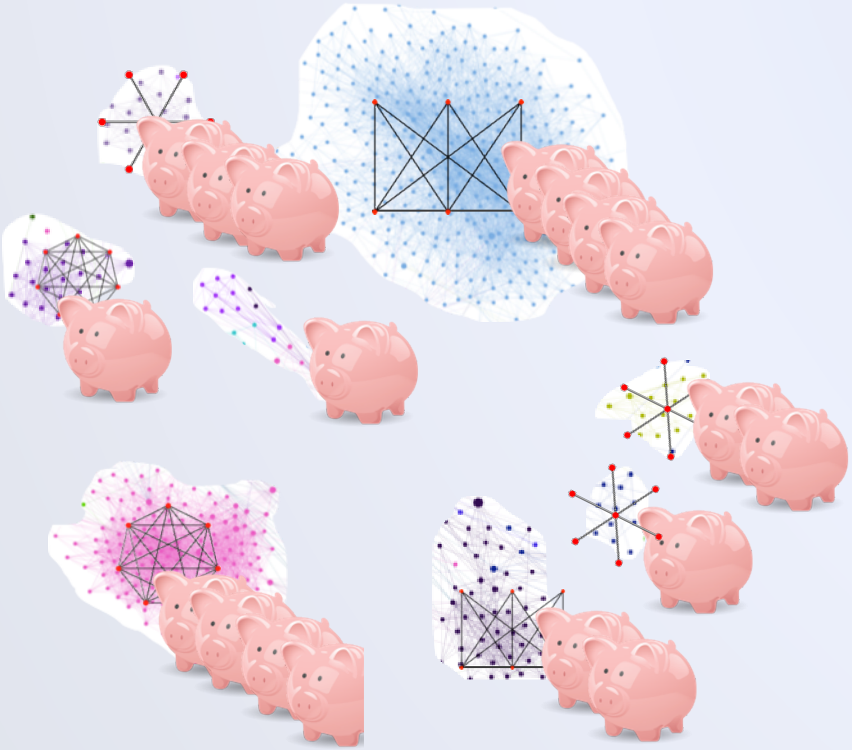
Graph Representations

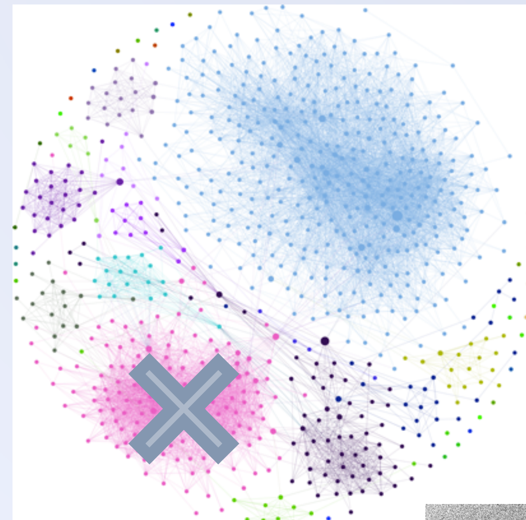


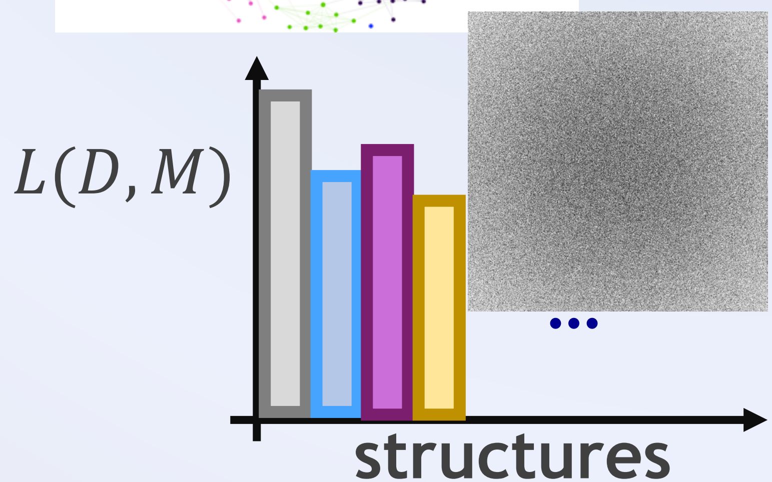
Step 3: Summary Assembly

Greedy&Forget

DETAILS

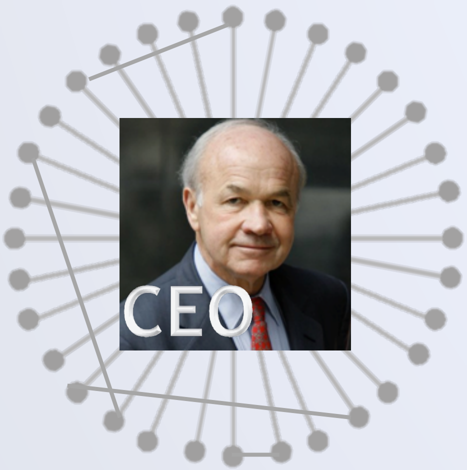






Qualitative Analysis: Enron

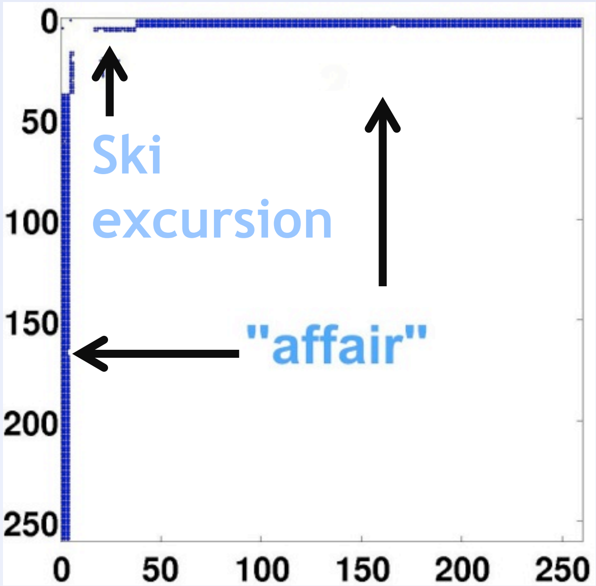
Top-3 Stars



klay
kenneth.lay@enron.com

jeff.skilling@enron.com

Top-1 NBC



Conclusions

we have shown

many **successful** applications

using *local* information based modelling
and information theory

clustering, outlier detection, data generation,
distance measures, missing value imputation,
change detection, privacy preservation,
graph clustering, influence propagation,
classification, …

all at an *explorative* angle

few assumptions and parameters
identify interesting **local** structure
describe structure in **simple** terms

Thank you!

we have shown

many **successful** applications

using local information based modelling
and information theory

clustering, outlier detection, data generation,
distance measures, missing value imputation,
change detection, privacy preservation,
graph clustering, influence propagation,
classification, …

all at an *explorative* angle

few assumptions and parameters
identify interesting **local** structure
describe structure in **simple** terms